C.U.F.R. J-F. Champollion — L2 Informatique S5
Albi — 2015

TP 7 (1 séance)
Plus courts chemins dans un graphe orienté valué

Le but de ce TP est de mettre en œuvre l'algorithme de Dijkstra pour la recherche des plus courts chemins dans un graphe orienté valué.

Le TP sera illustré à l'aide des graphes suivants :
G7, donné par la liste d'arêtes valuées :
{ {1,2,10},{1,3,3},{1,5,6},{2,1,0},{3,2,4}, {3,5,2},{4,3,1},{4,5,3},{5,2,0} ,{5,6,1},{6,1,2}, {6,2,1}}
Cyclique mais à poids positifs, il nous servira à illustrer l'algorithme de Dijkstra (version 1 mais les plus rapides pourront essayer de mettre en œuvre la version optimisé)

G8, donné par la liste d'arêtes valuées :
{1,2,1},{1,3,-2},{2,4,-2},{3,2,1},{3,4,5},{3,5,4},{5,6,-1},{6,4,-5}}
A poids négatifs mais sans cycle, il servira à illustrer l'algorithme de Bellman

Des illustrations de la mise en place de ces algorithmes sur G7 et G8 respectivement vous ont été fournies sur Moodle.

1. Classe GrapheOrienteValue

Créer la classe *GrapheOrienteValue* héritant de la classe *GrapheOrienteList* permettant de définir un graphe non orienté valué.

Copier et adapter les définitions d'attributs spécifiques de la classe *GrapheNonOrienteValue* :

private int[][] Poids; // La matrice des poids où P[i][j] est le poids de l'arc i->j
private ArrayList<int[]> ListeArete; // La liste des arêtes sous forme d'un triplet d'entiers {i,j,poids(i,j)}

Copier et adapter les constructeurs et les méthodes d'affichage :

- *public GrapheOrienteValue(int nb) :* construisant un graphe à nb sommets sans arêtes.
- *public GrapheOrienteValue(int nb,int[][] T) :* qui à partir d'un vecteur d'arêtes valuées de la forme {i,j,poids(i,j)} construise les attributs du graphe (donc ses attributs en tant que *GrapheOrienteList* plus ses attributs spécifiques en tant que *GrapheOrienteValue.*

- *public void affichePoids() :* qui affiche la matrice des poids
- *public static void afficheAretes(ArrayList<int[]> L)* qui affiche la liste des arêtes
- public void affiche() : qui appelle les deux affichages précédents.

2. Dijkstra

Compléter la méthode

- *public int[][] Dijkstra(int s)* : mettant en œuvre l'algorithme de Dijkstra pour la recherche des plus courts chemins dans un graphe orienté valué et retournant comme résultat le tableau {D,P}.