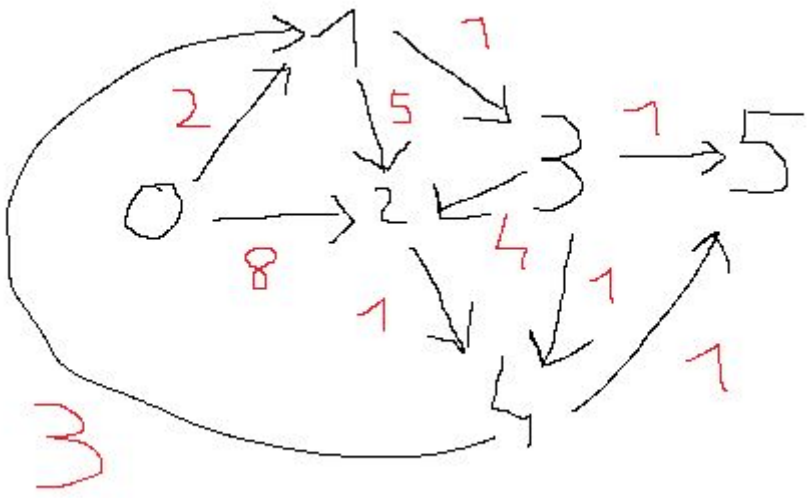
1) Implémentation de l'algorithme de Dijkstra

L'implémentation peut être trouvée dans le fichier "Dijkstra_dans_le_metro.c" sous le nom Dijkstra_exo1.

Elle est testée dans le fichier "main1.c" sur le graphe "graphe_test.graph" dont voici la représentation :



Le test effectué dans main.1 est fait à partir du sommet 0. Deux tableaux vont s'afficher : un premier répertoriant par ordre croissant de distance (calculées par Dijkstra) les sommets atteignables depuis le sommet 0 (en l'occurrence ici : tous); et un deuxième répertoriant toutes les distances d'éloignement pour chaque sommet du graphe, ces distances étant les plus courtes possibles calculées par l'algorithme de Dijkstra.

Il est bien entendu possible de modifier l'entier dans l'appel de la fonction de Dijkstra dans main1 pour observer le résultat depuis les autres sommets.

2) Algorithme des plus courts chemins.

L'implémentation peut être trouvée dans le fichier "Dijkstra_dans_le_metro.c" sous le nom PCC_exo2, elle est testée dans le fichier "main2.c" sur le même graphe, en cherchant le plus court chemin entre le sommet 1 et 4. Il est bien entendu possible de changer ces valeurs pour tester d'autres combinaisons.

L'algorithme en lui-même reprend l'algorithme de Dijkstra en créant un autre graphe avec le même nombre de sommets que celui étudié. En suivant l'algorithme de Dijkstra on ajoute successivement les arcs correspondant au plus court chemin (l'ajout d'arc est donc effectué à la fin de la boucle principale de l'algorithme de Dijkstra).

Dans main2.c, un appel à AfficheSuccesseurs est fait sur le graphe obtenu par l'algorithme pour visualiser le résultat de celui-ci.

3) Implémentation de l'algorithme de Dijkstra amélioré.

L'implémentation peut être trouvée dans le fichier "Dijkstra_dans_le_metro.c" sous le nom Dijkstra_exo3. Elle est testée dans le fichier main3.c sur le même graphe que précédemment en partant encore du sommet 0, pour permettre de comparer le résultat avec le 1er algorithme. Il est toujours possible de changer le sommet de départ dans la fonction main3.c

On utilise désormais deux tableaux de booléens U et T. U est initialisé rempli de TRUE et T de FALSE. Ainsi on ne vérifie plus la présence d'éléments dans S, mais les valeurs des booléens U[i] et T[i] selon l'endroit où l'on se trouve dans l'algorithme.

Avoir une distance à un sommet non infinie fait changer le booléen correspondant dans U à FALSE et celui de T à TRUE.

A chaque fois qu'un sommet intègre S, les booléens correspondants dans U et T sont automatiquement passés à FALSE. L'algorithme s'arrête toujours une fois avoir effectué n itérations de la boucle principale, mais nécessite des tests beaucoup plus courts que dans la 1ere version.

NB : Pour de plus amples informations sur la manipulation des algorithmes, merci de se référer au fichier README présent dans l'archive.