Tolkacheva Anastasia

Schaeffer Johan

**Rapport**

**Algorithmes de parcours de graphes**

======================================================================

**2 Représentation d’un graphe (2h)**

Pour représenter un graphe orienté, nous avons construit plusieurs classes en Java. Chaque nœud du graphe est identifié par une chaîne de caractères (par exemple "A" ou "B") et possède une liste de nœuds adjacents, c’est-à-dire les nœuds vers lesquels un arc le relie.

Classe Arc

Question 1 :

Nous avons créé une classe Arc pour représenter un arc. Cet arc contient deux informations : le nom du nœud de destination (une chaîne de caractères) et le coût de l’arc (un nombre réel positif). Cela nous permet de connaître où va l’arc et combien il coûte.

Classe Arcs

Question 2 :

Ensuite, nous avons fait une classe Arcs pour regrouper plusieurs arcs qui partent d’un même nœud. Cette classe utilise une liste d’objets Arc. Elle permet d’ajouter un arc et de récupérer la liste complète des arcs partants.

Interface Graphe

Question 3 :

Pour généraliser la structure d’un graphe, nous avons écrit une interface Graphe qui impose deux méthodes :

* listeNoeuds() pour avoir la liste de tous les nœuds du graphe ;
* suivants(String n) pour récupérer tous les arcs partant d’un nœud donné.

Classe GrapheListe

Question 4 :

Nous avons ensuite réalisé une classe GrapheListe qui implémente l’interface Graphe. Cette classe utilise deux listes : une pour les noms des nœuds, et une autre pour leurs arcs. L’index dans ces deux listes permet de relier un nœud à ses arcs.

Nous avons aussi codé une méthode ajouterArc(String depart, String destination, double cout) qui permet d’ajouter un nouvel arc dans le graphe. Si les nœuds n’existent pas encore, ils sont ajoutés automatiquement.

Question 5 :

Dans une méthode main, nous avons recréé le graphe de l’exemple donné dans le sujet. Cela nous a permis de tester l’ajout des arcs et de vérifier le bon fonctionnement de notre structure.

Question 6 :

Pour afficher le graphe, nous avons écrit une méthode toString() dans la classe GrapheListe. Elle affiche chaque nœud suivi de ses arcs sous forme :

A-> B(12) D(87)

B-> E(11)

C-> A(19)

D-> B(23) C(10)

E-> D(43)

Question 7 :

Enfin, nous avons écrit des tests unitaires pour vérifier que le graphe est bien construit. Nous avons vérifié la liste des nœuds, les arcs de chaque nœud et l’affichage avec toString().