École de technologie supérieure	**Trimestre**	**Hiver 2016**
Département de génie logiciel et des TI	**Professeur**	**Stéphane Coulombe**
	Préalable	**LOG120**
	Crédits	**4**

GTI310 Structure de données multimédias

Troisième Travail Pratique

Sujet : La théorie des graphes

Description

La compagnie Unreal-Networks International œuvre dans l'optimisation des processus. Elle agit autant au niveau des routes de livraison qu'au niveau de la transmission de données informatiques.

La compagnie Chasse-Neige, une entreprise de déneigement, a contacté Unreal-Networks International pour optimiser ses routes de déneigement. La compagnie aimerait éviter de passer par les rues déjà déneigées de façon à diminuer la quantité d'essence utilisée ainsi que de réduire le temps mis pour déneiger les rues de la ville Grosse-Neige.

Unreal-Networks International vous a mandaté pour développer une application permettant de résoudre le problème de la compagnie Chasse-Neige.

L'application devra être en mesure d'indiquer tous les chemins que peut emprunter la déneigeuse de façon à ce qu'elle revienne à son point de départ en empruntant chaque rue une seule fois par direction.

S'il n'est pas possible que chaque rue soit empruntée une seule fois, la compagnie se contentera d'obtenir un message l'informant qu'il n'existe pas de chemin parfait.

Votre application sera utilisée pour résoudre ce problème, mais elle devra être capable de résoudre n'importe quel problème. Il est possible que Grosse-Neige construise de nouvelles rues dans le futur. De plus, votre application devra gérer les rues à sens unique et à double sens.

Voici un plan des rues de la ville Grosse-Neige :

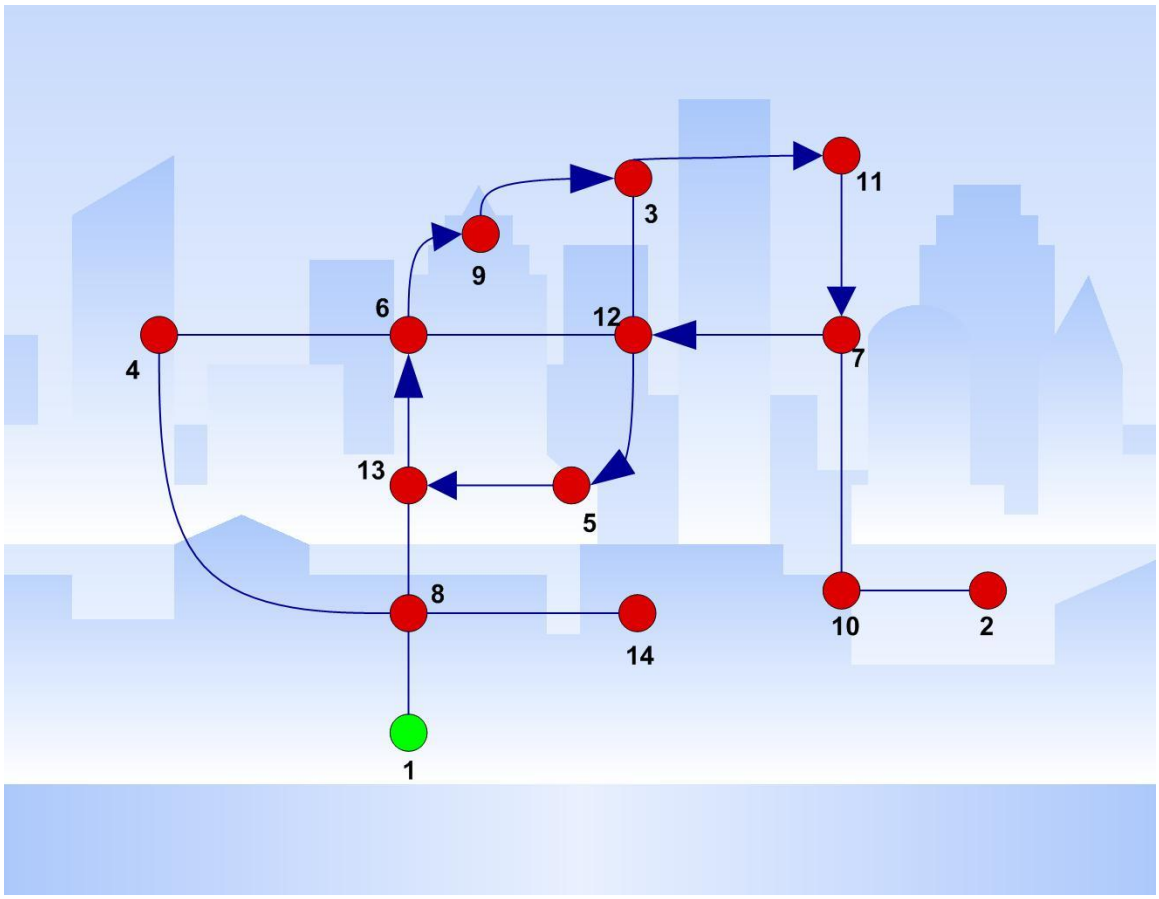

Figure 1 Plan des rues de Belle-Neige

La compagnie de déneigement Chasse-Neige commence le déneigement au point 1 (le point vert).

Exigences

Architecture

Les diagrammes ci-dessous devraient vous aider à mieux comprendre l'architecture à utiliser. Une liste des tâches à compléter est aussi présentée :

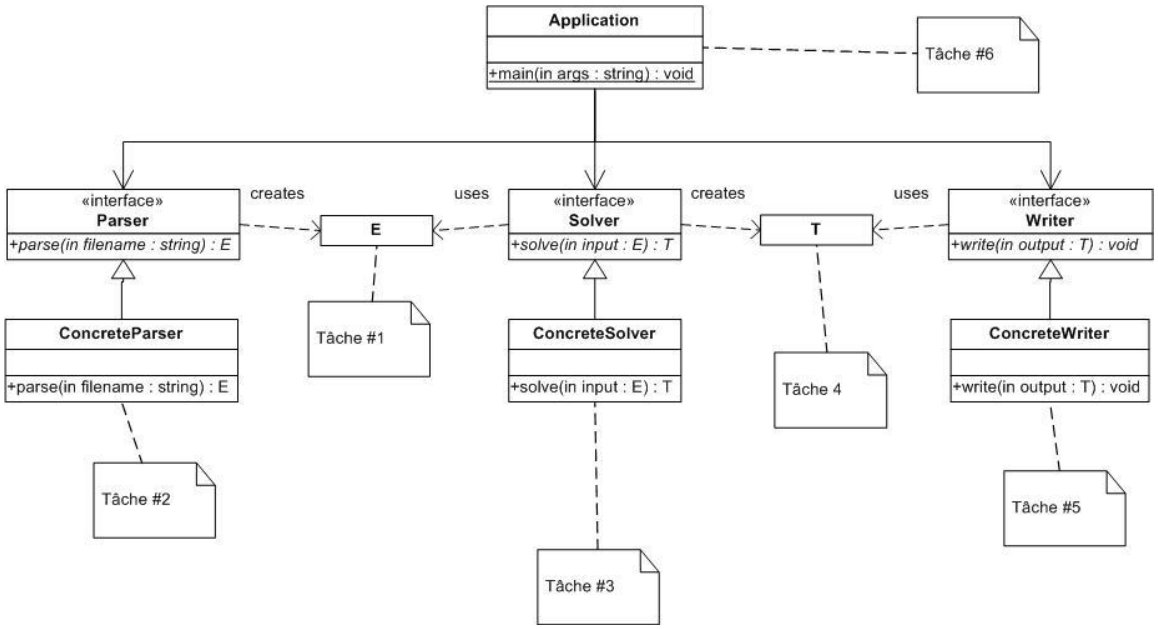

Figure 2 Diagrammes de classes

Les tâches à compléter:

1. Créer un objet (une classe) pour stocker l'information extraite du fichier à l'entrée. L'objet servira à la prochaine étape, donc il serait intelligent de prévoir des méthodes d'accès pour récupérer l'information.
2. Implémenter votre interpréteur de fichier source. Vous devrez utiliser la classe que vous avez créée à l'étape précédente. Lisez bien les commentaires dans le fichier « Parser.java » pour bien comprendre comment construire et initialiser votre objet.
3. Créer un objet (une classe) pour stocker l'information produite par la classe servant à résoudre le problème. Cette classe servira à la toute fin pour inscrire votre réponse dans un fichier de sortie.
4. Implémenter une classe concrète servant à résoudre la problématique. Vous devrez vous servir des classes créées aux étapes 1 et 3. Lisez attentivement les commentaires dans le fichier « Solver.java ». Ils vous expliqueront comment créer votre classe et comment l'instancier par la suite.
5. Bâtir une classe concrète pour écrire votre solution dans un fichier. La classe créée à l'étape 3 devra être utilisée pour récupérer la solution. Vous devrez respecter le format de sortie spécifié.
6. Ajouter le code nécessaire à la classe « Application » pour qu'elle appelle séquentiellement vos outils. Le diagramme de séquence ci-dessous présente un sommaire des interactions dans le logiciel :

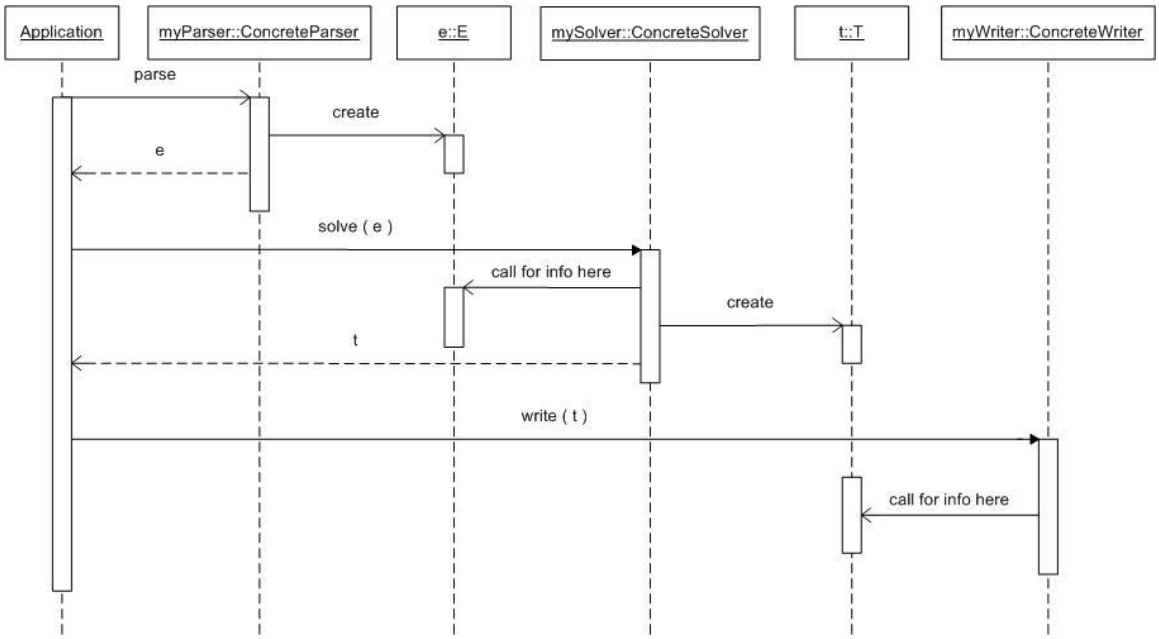

Figure 3 Diagramme de séquence

Fichiers

Les données du problème vont être fournies sous forme de fichier texte. Vous devrez décortiquer le fichier pour obtenir l'information nécessaire (point de départ, nombre de sommets, etc.). À partir de ces informations, vous devrez bâtir une matrice de poids pour résoudre le problème (et d'autres problèmes du même type).

```
<nombre de sommets>
<valeur pour l'infinie>
<sommet de départ>
<source><tabulation><destination><tabulation><poids>
<source><tabulation><destination><tabulation><poids>
…
<source><tabulation><destination><tabulation><poids>
$
```

Le symbole « $ » marquera la fin du fichier. Il se trouvera seul sur une ligne.

Votre application devra traiter le cas où un fichier invalide a été fourni à l'entrée. Il ne faudra pas que l'application continue si les données à la source ne sont pas correctes.

Il est possible qu'il n'y ait pas de sommet de départ indiqué. Dans un tel cas, la valeur 0 devra être utilisée : cette valeur devrait correspondre au premier point dans la matrice de poids.

Votre solution sera ensuite écrite dans un fichier texte. Il sera important de respecter le format de sortie. Ce dernier devrait avoir l'allure suivante :

```
<départ><espace><sommet><espace><sommet>…<départ>
<départ><espace><sommet><espace><sommet>…<départ>
<départ><espace><sommet><espace><sommet>…<départ>
…
```

Chaque cycle commence et finit au même point. Chaque cycle est inscrit sur une seule ligne.

```
0 -> 1 -> 3 -> 2 -> 5 -> 7 -> 6 -> 4 -> 0
0 -> 2 -> 3 -> 1 -> 7 -> 5 -> 6 -> 4 -> 0
0 -> 2 -> 3 -> 1 -> 7 -> 6 -> 4 -> 5 -> 0
0 -> 2 -> 3 -> 1 -> 7 -> 6 -> 5 -> 4 -> 0
```

Exécution

Votre programme devra être appelé de la console. Deux paramètres sont exigés : le nom du fichier à l'entrée et le nom du fichier à la sortie.

```
java <programme> <ficher d'entrée> <fichier de sortie>
```

Analyse de complexité

Vous devrez analyser (notation O) la complexité des méthodes de votre classe « ConcreteSolver ». Cette analyse devra se trouver dans les commentaires de l'en-tête des méthodes servant à la résolution de problème.

À remettre

- Un rapport de laboratoire
  - Voir le gabarit qui est disponible sur le site du cours.
- Le projet contenant votre code source.
  - **Ne pas remettre les fichiers texte utilisés ou produits.**

Critères d'évaluation

1. Rapport de laboratoire : 40 %
2. Programme : 60 %
   a. Qualité du code : 15 %
   b. Fonctionnement : 40 %
   c. Analyse de complexité : 5 %

Date de remise

Au plus tard le 17 mars 2016, au début du laboratoire. Votre travail devra être remis électroniquement au chargé de laboratoire.

Pénalité de retard

Les dates de remises des travaux doivent être respectées. Une pénalité de 10 % par jour ouvrable sera appliquée aux travaux qui ne sont pas remis à temps.

Plagiat et fraude

Les clauses du « Chapitre 10 : Plagiat et fraude » du « Règlement des études de 1er cycle » s'appliquent dans ce cours ainsi que dans tous les cours du département de génie logiciel et des TI.