Travail pratique #3 : Planification d'un réseau de panneaux publicitaires électroniques

1. Objectifs

- Implémenter un type abstrait de données `Carte` basé sur la représentation de graphe.
- Implémenter un algorithme de recherche d'un arbre de recouvrement.

2. Problématique

Vous devez écrire un programme en Java nommé `Tp3`.
Le programme `Tp3` doit chercher un arbre de recouvrement d'un graphe.

La mairie d'une ville a décidé d'installer des terminaux publicitaires dans un quartier. Elle veut relier tous les sites (places publiques) du quartier entre eux via un réseau filaire. Elle vous a fourni les données suivantes :

- La liste des rues les plus visitées du quartier;
- Le cout associé à l'installation des panneaux sur un segment de la rue.

Chaque rue est identifiée par un ensemble de sites. Les rues et les sites sont identifiés par les noms : **`rue0 a b`**.

`rue0` – nom de la rue; **`a`**, **`b`** – les noms des sites.

Exemple : **`rue0 a b 4`**

Les sites du quartier sont reliés entre eux en formant un graphe connexe étiqueté où les nœuds représentent les sites du quartier, les arêtes sont les segments des rues qui relient les sites entre eux et les poids sont le cout sur chaque segment. La ville veut relier, de façon optimale, tous les sites du cartier par un réseau publicitaire en diminuant les dépenses d'installation. L'objectif du TP3 est de fournir un arbre de recouvrement à coût minimum (ARMin) du graphe des sites d'un quartier en se basant sur l'algorithme de Kruskal. Les arêtes de mêmes poids doivent être traitées selon l'ordre alphanumérique des nœuds de départs et dans le cas d'égalité des poids et des nœuds de départs, utiliserez l'ordre alphanumérique des nœuds d'arrivés.

3. Structure de programme

Pour la correction automatique, vous devez préserver la syntaxe d'appel du programme et ses formats d'entrée et de sortie.

3.1 Syntaxe d'appel

Le programme `Tp3` doit pouvoir être lancé en ligne de commande avec la syntaxe suivante.

```
java Tp3 carte.txt arm.txt
```

Le fichier `carte.txt` spécifie une carte des sites d'un cartier et les données de coût associées à l'installation de panneaux sur chaque rue.

Les résultats produits par votre programme doivent être écrits dans le fichier `arm.txt`

3.2 Fichier `carte.txt`

Le fichier **`carte`**`.txt` est constitué de :

1. Une liste des sites (nœuds). Un site est spécifié par un nom (une chaîne de caractères);
2. Trois tirets (---) de séparation.
3. Une liste des rues d'installation de panneaux.
   Une rue est spécifiée par un nom de rue (une chaîne de caractères); un deux-points (:); une paire de sites représentant un point de départ et un point d'arrivé d'une arête; un nombre caractérisant le coût d'installation sur ce segment de la rue (arête); un point-virgule (;).
4. Trois tirets (---) de fin.

À titre d'exemple, voici le fichier (**`carte0.txt`**) :

```
a
b
c
d
e
f
g
h
i
---
rue0 : a b 4 ;
rue1 : a h 8 ;
rue2 : b h 11 ;
rue3 : b c 8 ;
rue4 : c i 2 ;
rue5 : c f 4 ;
rue6 : c d 7 ;
rue7 : d f 14 ;
```

```
rue8 : d e 9 ;
rue9 : e f 10 ;
rue10 : f g 2 ;
rue11 : g i 6 ;
rue12 : g h 1 ;
rue13 : h i 7 ;
---
```

3.3 Format de sortie pour TP3

Le programme TP3 doit afficher l'arbre ARM de la façon suivante.

Afficher d'abord les sites, un nœud par ligne. Ensuite, afficher les arêtes en ordre croissant alphanumérique des nœuds de départs. Si plusieurs arêtes ont le même nœud de départ, vous devez afficher ces arêtes selon l'ordre croissant alphanumérique des nœuds d'arrivés.

Pour chaque arête vous devez afficher sur une ligne:

a) le nom de la rue : `rue0`
b) les sommets d'une arête : `a b`
c) une étiquette : `4`

Après avoir affiché toutes les arrêtes :
1. Trois tirets (---) de séparation;
2. Le coût total de l'arbre sur la dernière ligne.

Exemple de sortie.

```
a
b
c
d
e
f
g
h
i
rue0 a b 4
rue1 a h 8
rue6 c d 7
rue4 c i 2
rue8 d e 9
rue5 f c 4
rue10 g f 2
rue12 h g 1
---
37
```

5. Tests

La correction sera faite de manière semi-automatique, donc le respect de la syntaxe d'appel et de la sortie est obligatoire. La note sera basée sur les résultats de passage de nos tests.

6. Remise

Vous devez remettre le TP3 au plus tard le 28 juillet à 23h59. Il y aura une pénalité de **10%** par jour de retard avec 3 jours de retard maximum.

6.1 Rapport

Une analyse de la complexité algorithmique de votre solution.

- Identifiez les principaux facteurs qui influencent le temps d'exécution. Indice : la taille du problème.
- Donnez la complexité temporelle en notation grand O. Avec une justification.
- **Le rapport doit être remis électroniquement sur StudiUM.**

6.2 Remise électronique du code source

StudiUM

---

7. ÉVALUATION

Ce travail pratique vaut 10% de la note finale.

Grille de correction

Critère	Description	Pondération
A.	**Respect des directives pour la remise**	/ 1
B.	**Appréciation générale** Structure du programme + Qualité du code : découpage du programme, choix des types de données; identificateurs (noms) significatifs, lisibilité du code, pertinence des commentaires; etc. Encapsulation : respect des principes de l'abstraction; cachez le maximum de la représentation des objets en rendant un maximum d'attributs privés;	/ 2
C.	**Fonctionnement correct.** Le programme produit les bons résultats. L'efficacité n'est pas directement évaluée. Cependant, l'efficacité peut être indirectement évaluée lorsqu'un	/ 4

	programme ne parvient pas à produire des résultats dans des délais raisonnables.	
D.	**Exactitude de l'auto-évaluation.** Vous déclarez que votre programme fonctionne correctement, partiellement, aucunement**.**	/ 0.5
E.	**Analyse de l'algorithme**	/ 2.5
	Total :	10/ 10

Pour les cas problématiques, jusqu'à 2 points peuvent être retranchés pour la qualité de la présentation.