UNIVERSITÉ LAVAL
Faculté des sciences et de génie
Département d'informatique et de génie logiciel

DÉPARTEMENT D'INFORMATIQUE et de GÉNIE LOGICIEL

IFT-2008 – Algorithmes et structures de données

Été 2025

Travail pratique 2 (individuel)

À remettre, par le portail du cours, avant 17h00 le dimanche 6 juillet 2025

---

1 Mise en contexte

Ce projet vise à implémenter un réseau de bornes de recharge pour véhicules électriques, modélisé comme un **graphe orienté, pondéré et étiqueté**. Chaque sommet représente une borne de recharge, chaque arc un trajet entre deux bornes. Chaque trajet est caractérisé par trois pondérations principales :

- **Distance** : en kilomètres.
- **Temps de recharge** : en minutes, à la borne d'arrivée.
- **Coût énergétique** : en dollars, pouvant être négatif.

L'objectif du système est de permettre :

- **Recherche de chemin optimal** : déterminer selon le critère choisi (distance, temps ou coût) le parcours le plus avantageux entre deux bornes ;
- **Prise en charge de graphes complexes** : gérer cycles et coûts négatifs grâce à Bellman-Ford.

2 Modélisation du système

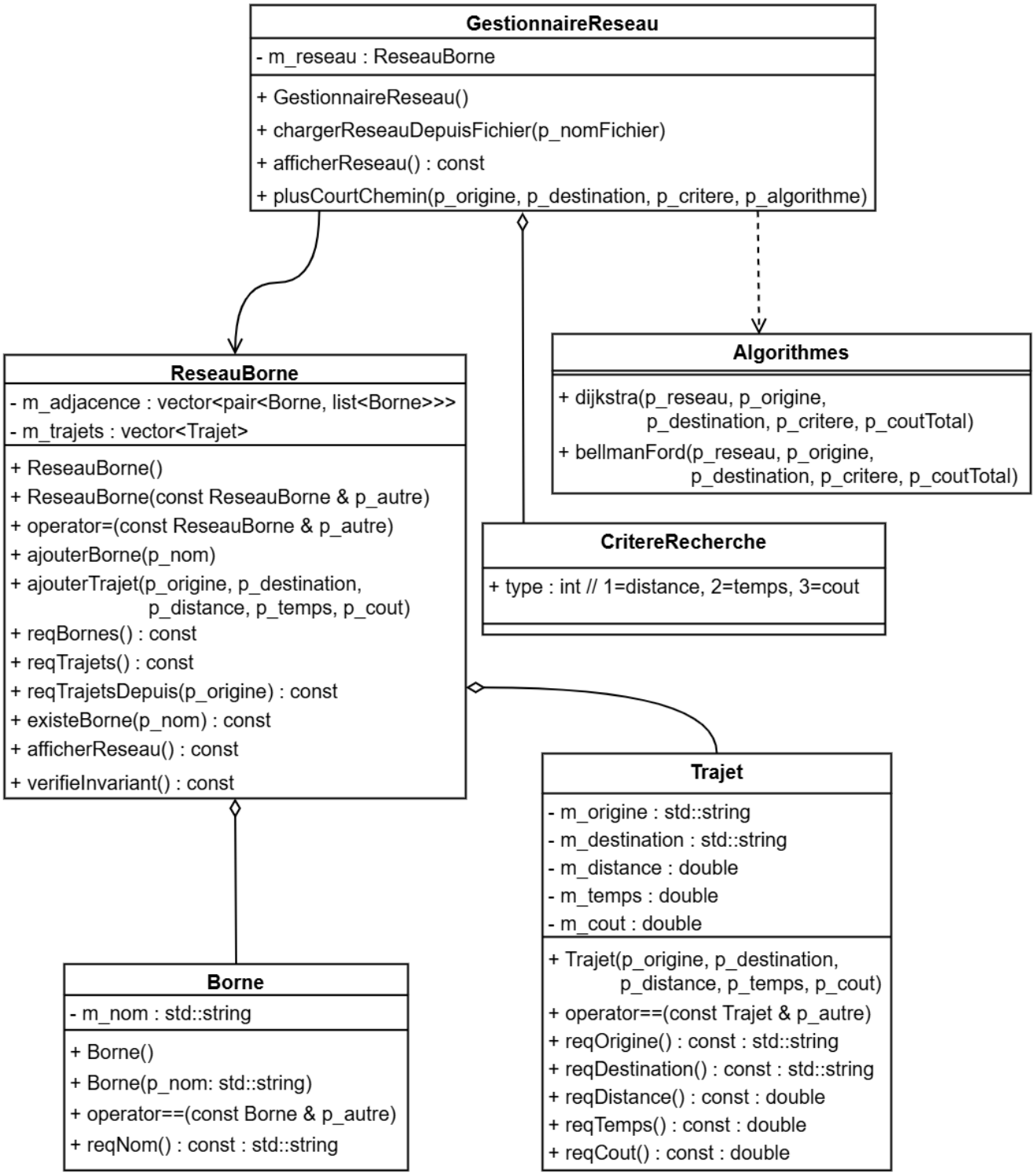


Figure 1 – Diagramme de classes UML du TP

3 Fichiers Sources

- `Principal.cpp` : Point d'entrée, gestion de la boucle principale, du menu et des interactions utilisateur.
- `ContratException.cpp`, `ContratException.h` : Gestion des exceptions et de la théorie du contrat (précondition, invariant, etc).
- `GestionnaireReseau.cpp`, `GestionnaireReseau.h` : Orchestration des opérations haut niveau sur le réseau, appels aux algorithmes et gestion des scénarios.
- `ReseauBorne.cpp`, `ReseauBorne.h` : Modélisation du graphe des bornes et trajets.
- `Borne.cpp`, `Borne.h` : Représentation d'une borne de recharge.
- `Trajet.cpp`, `Trajet.h` : Représentation d'un trajet orienté entre deux bornes avec pondérations.
- `Algorithmes.cpp`, `Algorithmes.h` : Implémentation des algorithmes de graphe (Dijkstra, Bellman-Ford).

4 Fichiers de Données

- `data/ReseauBorne.txt` : Fichier texte contenant la définition du réseau canadien avec 13 bornes et 23 trajets.

5 Visualisation du réseau complet

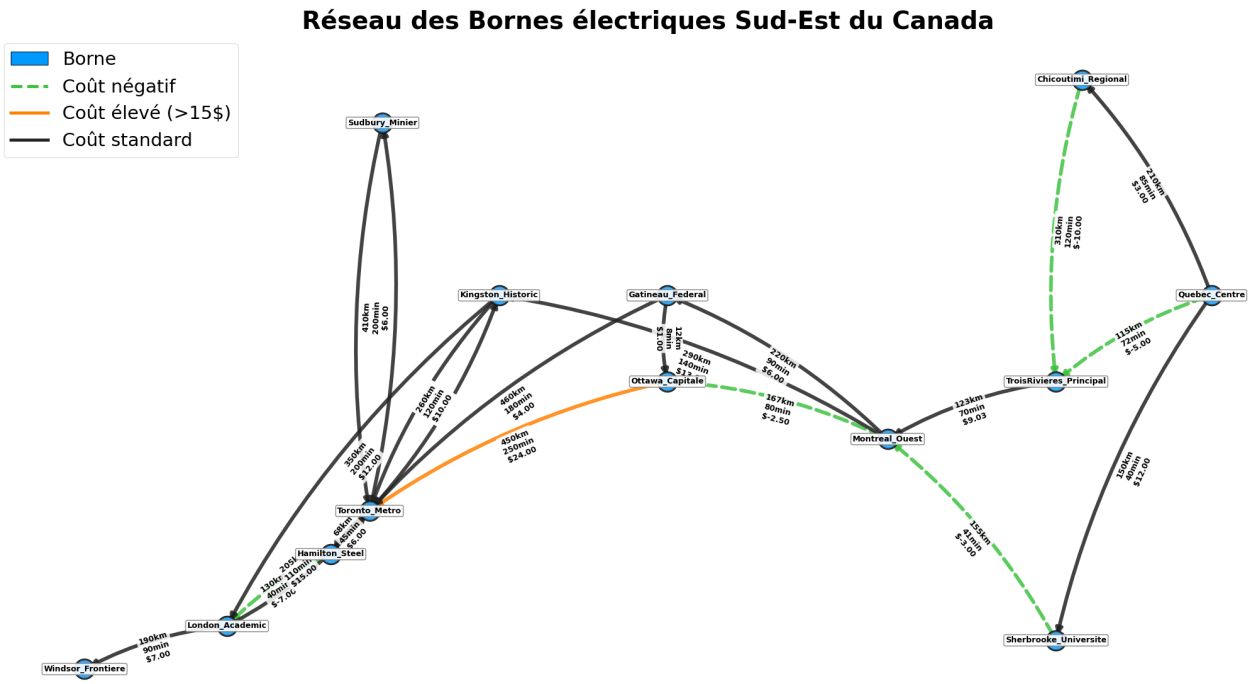
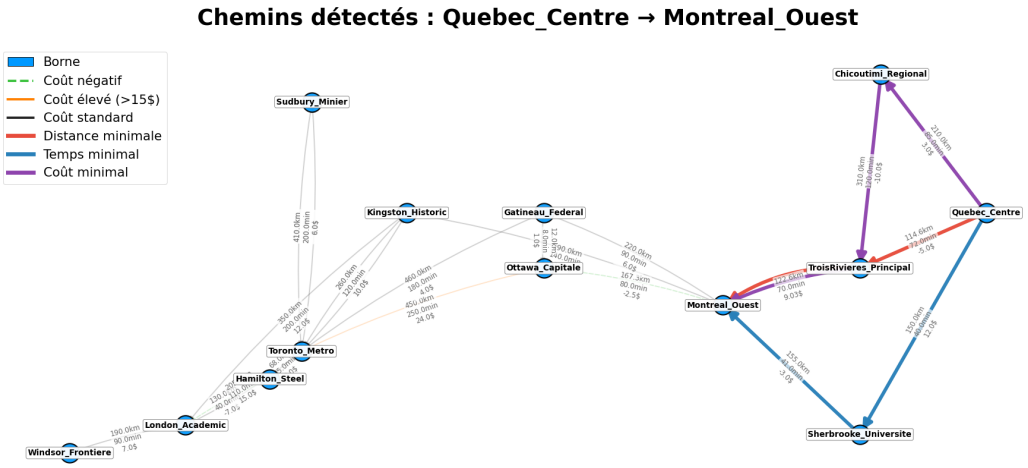


Figure 2 – Topologie complète du réseau de bornes électriques et leurs connexions.

6 Exemples de scénarios optimaux (analyse graphique)

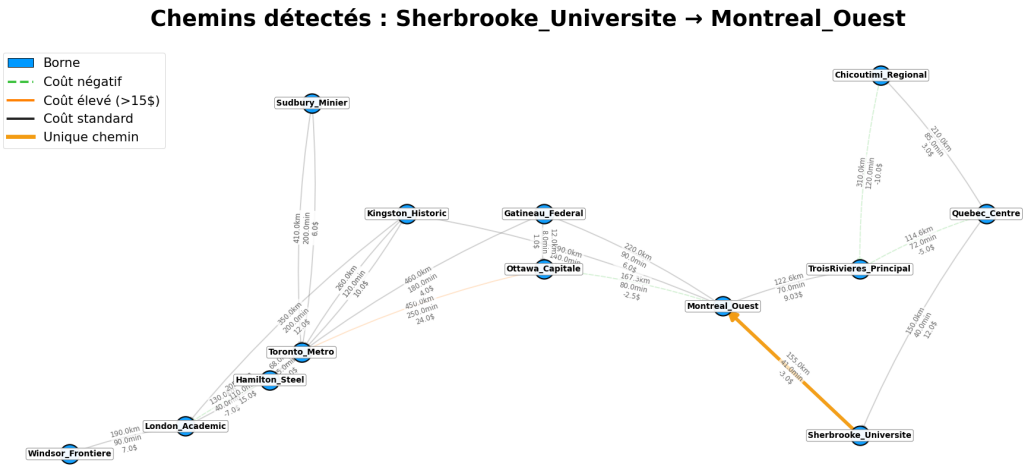
Exemple 1 : Québec Centre → Montréal Ouest (distance, temps, coût)



Critère	Bornes traversées	Distance (km)	Temps (min)	Coût ($)
Distance minimale	Quebec_Centre → TroisRivieres_Principal → Montreal_Ouest	237.20	142.00	4.03
Temps minimal	Quebec_Centre → Sherbrooke_Universite → Montreal_Ouest	305.00	81.00	9.00
Coût minimal	Quebec_Centre → Chicoutimi_Regional → TroisRivieres_Principal → Montreal_Ouest	642.60	275.00	2.03

Figure 3 – Comparaison des chemins optimaux entre Québec Centre et Montréal Ouest.

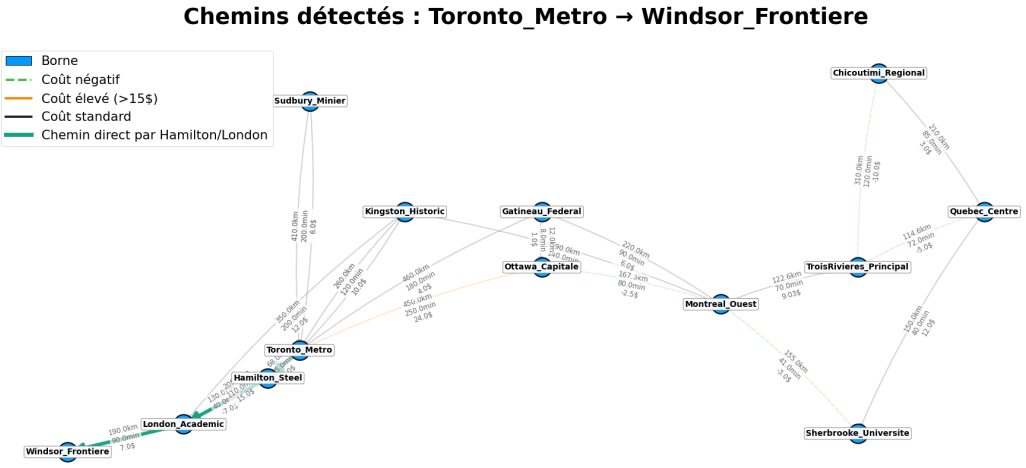
Exemple 2 : Sherbrooke Université → Montréal Ouest



Critère	Bornes traversées	Distance (km)	Temps (min)	Coût ($)
Unique chemin	Sherbrooke_Universite → Montreal_Ouest	155.00	41.00	-3.00

Figure 4 – Chemin optimal entre Sherbrooke Université et Montréal Ouest.

Exemple 3 : Toronto Metro → Windsor Frontière



Critère	Bornes traversées	Distance (km)	Temps (min)	Coût ($)
Chemin direct par Hamilton/London	Toronto_Metro → Hamilton_Steel → London_Academic → Windsor_Frontiere	388.00	175.00	6.00

Figure 5 – Chemin optimal entre Toronto Metro et Windsor Frontière.

7 Description détaillée des classes

Borne

Classe représentant une station de recharge.

- **Attributs** :
  - `m_nom` : nom unique de la borne
- **Méthodes** :
  - `Borne()` : constructeur par défaut
  - `Borne(p_nom)` : constructeur initialisant le nom (précondition : non vide)
  - `operator==(const Borne& p_autre) const` : comparaison de deux bornes
  - `reqNom()` : accesseur au nom

Trajet

Classe représentant un trajet orienté entre deux bornes avec des pondérations.

- **Attributs** :
  - `m_origine` : nom de la borne de départ
  - `m_destination` : nom de la borne d'arrivée
  - `m_distance` : distance en kilomètres
  - `m_temps` : temps de recharge en minutes
  - `m_cout` : coût énergétique en dollars

- **Méthodes** :
  - `Trajet(p_origine, p_destination, p_distance, p_temps, p_cout)` : constructeur (préconditions : noms non vides, distance/temps 0)
  - `operator==(const Trajet& p_autre) const` : comparaison complète
  - `reqOrigine()` : accesseur à l'origine
  - `reqDestination()` : accesseur à la destination
  - `reqDistance()` : accesseur à la distance
  - `reqTemps()` : accesseur au temps de recharge
  - `reqCout()` : accesseur au coût énergétique

ReseauBorne

Classe qui modélise le graphe des bornes et trajets.

- **Attributs** :
  - `m_adjacence` : (vector<pair<Borne, list<Borne»>) liste d'adjacence, chaque Borne et ses voisins accessibles
  - `m_trajets` : (vector<Trajet>) collection de tous les trajets du réseau
- **Méthodes** :
  - `ReseauBorne()` : constructeur par défaut
  - `ReseauBorne(const ReseauBorne&)` : constructeur de copie
  - `operator=(const ReseauBorne&)` : assignation
  - `ajouterBorne(p_nom)` : ajoute une borne au réseau
  - `ajouterTrajet(...)` : ajoute un trajet
  - `reqBornes()` : retourne toutes les bornes
  - `reqTrajets()` : retourne tous les trajets
  - `reqTrajetsDepuis(p_origine)` : retourne les trajets partant d'une borne
  - `existeBorne(nom)` : Vérifie si une borne existe
  - `afficherReseau()` : affiche le graphe dans le CLI
  - `verifieInvariant()` : appelée par les macros `INVARIANTS` pour s'assurer que chaque trajet pointe vers des bornes existantes et que la table d'adjacence reste cohérente

GestionnaireReseau

Gestion des opérations haut niveau sur le réseau.

- **Attributs** :
  - `m_reseau` : instance du réseau de bornes
- **Méthodes** :
  - `GestionnaireReseau()` : constructeur
  - `chargerReseauDepuisFichier(p_nomFichier)` : charge le réseau depuis un fichier texte
  - `afficherReseau()` : affiche la topologie du réseau
  - `plusCourtChemin(p_origine, p_destination, critere, algorithme)` : cherche le plus court chemin

CritereRecherche

Structure pour transmettre le critère de recherche (dans `GestionnaireReseau.h`).

- **Attributs** :
  - `type` : (int) 1 pour distance, 2 pour temps, 3 pour coût
- **Utilisation** :
  - Sert à transmettre le critère lors du calcul d'un chemin optimal

Algorithmes

Classe utilitaire statique pour l'optimisation et l'analyse.

- **Méthodes statiques** :
  - `dijkstra(p_reseau, p_origine, p_destination, p_critere, p_coutTotal)` : plus court chemin selon Dijkstra
  - `bellmanFord(p_reseau, p_origine, p_destination, p_critere, p_coutTotal)` : plus court chemin selon Bellman-Ford

ContratException

Classe centrale pour la gestion robuste des erreurs liées aux contrats dans le système.

- **Classes dérivées :**
  - `AssertionException` : gestion des erreurs d'assertion
  - `PreconditionException` : gestion des erreurs de précondition
  - `PostconditionException` : gestion des erreurs de postcondition
  - `InvariantException` : gestion des erreurs d'invariant
- **Attributs :**
  - `m_expression` : expression logique qui a échoué
  - `m_fichier` : nom du fichier où a eu lieu l'erreur
  - `m_ligne` : numéro de la ligne d'erreur
- **Méthodes :**
  - `reqTexteException()` : retourne un message d'erreur formaté
- **Utilisation :**
  - Ces exceptions sont déclenchées grâce aux macros suivantes, à insérer dans chaque méthode concernée :
    - `ASSERTION`
    - `PRECONDITION`
    - `POSTCONDITION`
    - `INVARIANTS`
  - Ces vérifications garantissent la robustesse du système et sont obligatoires dans toutes les méthodes où des invariants ou des préconditions existent.

Principal

Classe de la boucle principale et du menu utilisateur.

- **Méthodes** :
  - `main()` : lance l'application, boucle sur le menu principal
  - `afficherMenu()` : affiche toutes les options du menu utilisateur
  - `clearInput()` : vide le tampon d'entrée pour éviter les erreurs de saisie
  - `demanderBorne()` : demande à l'utilisateur de saisir le nom d'une borne
  - `demanderCritere()` : demande à l'utilisateur de choisir un critère de recherche

8 Organisation et logique du menu (`Principal.cpp`)

Les points importants de l'organisation sont :

- À chaque sélection, l'option choisie est explicitement affichée à l'utilisateur.
- Pour la recherche de chemin, toutes les entrées nécessaires (bornes d'origine, destination, critère) sont demandées à l'utilisateur dans le menu, affichées, puis transmises au gestionnaire.
- Deux options distinctes existent pour la recherche de chemin :
  - Recherche de chemin avec Dijkstra (critère autorisé : distance ou temps, coût interdit)
  - Recherche de chemin avec Bellman-Ford (critère autorisé : distance, temps, ou coût, y compris coût négatif)
- Tous les appels au gestionnaire passent les valeurs saisies par l'utilisateur, chaque input étant affiché pour transparence.

9 Travail à faire

Il s'agit essentiellement d'implémenter dans les fichiers `Algorithmes.cpp`, `Borne.cpp`, `Trajet.cpp` et `ReseauBorne.cpp` les méthodes des classes définies dans les fichiers .h. Afin de vous aider, nous vous fournissons un code qui compile, mais qui ne donne pas évidemment le bon résultat.

Fichiers déjà complets – NE PAS MODIFIER

Les fichiers suivants sont déjà entièrement implémentés et utilisés par nos tests internes. Toute modification autre que de la documentation (commentaires Doxygen) risque de briser l'automatisation de la correction :

- `GestionnaireReseau.cpp` et `GestionnaireReseau.h`
- `Principal.cpp` (boucle de menu et entrées utilisateur)
- `ContratException.cpp` et `ContratException.h`
- La fonction `afficherReseau()` fournie dans `ReseauBorne.cpp` (identifiée par un encadré NE PAS MODIFIER).

Nous vous fournissons également un fichier Principal.cpp comprenant un main() qui vous permettra de vous aider à tester votre système. Nous avons déjà mis quelques exemples dans le

main, mais il en faut normalement d'autres. Par ailleurs, voici ce que le fichier Principal.cpp est censé donner comme résultat dans la console de la VM :

```
+=======================================================+
|  SIMULATEUR DE RÉSEAU DE BORNES ÉLECTRIQUES           |
+=======================================================+

+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 1

[INITIALISATION]
  > Fichier : data/ReseauBorne.txt
Chargement du réseau effectué avec succès !
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 2
[AFFICHAGE] État actuel du réseau :

========== TOPOLOGIE DU RÉSEAU ==========
Borne : Borne_London_Academic
    -> Borne_Windsor_Frontiere | Distance : 190 km | Temps : 90 min |
       Coût : 7 $
    -> Borne_Toronto_Metro | Distance : 205 km | Temps : 110 min | Coût
        : 15 $
Borne : Borne_Kingston_Historic
    -> Borne_Toronto_Metro | Distance : 260 km | Temps : 120 min | Coût
        : 10 $
    -> Borne_London_Academic | Distance : 350 km | Temps : 200 min |
       Coût : 12 $
Borne : Borne_Hamilton_Steel
    -> Borne_London_Academic | Distance : 130 km | Temps : 40 min |
       Coût : -7 $
```

```
Borne : Borne_Gatineau_Federal
    -> Borne_Ottawa_Capitale | Distance : 12 km | Temps : 8 min | Coût : 1 $
    -> Borne_Toronto_Metro | Distance : 460 km | Temps : 180 min | Coût : 4 $
Borne : Borne_Chicoutimi_Regional
    -> Borne_TroisRivieres_Principal | Distance : 310 km | Temps : 120 min | Coût : -10 $
Borne : Borne_Toronto_Metro
    -> Borne_Hamilton_Steel | Distance : 68 km | Temps : 45 min | Coût : 6 $
    -> Borne_Sudbury_Minier | Distance : 410 km | Temps : 200 min | Coût : 8 $
    -> Borne_Kingston_Historic | Distance : 260 km | Temps : 120 min | Coût : 10 $
Borne : Borne_Sherbrooke_Universite
    -> Borne_Montreal_Ouest | Distance : 155 km | Temps : 41 min | Coût : -3 $
Borne : Borne_Sudbury_Minier
    -> Borne_Toronto_Metro | Distance : 410 km | Temps : 200 min | Coût : 6 $
Borne : Borne_Montreal_Ouest
    -> Borne_Ottawa_Capitale | Distance : 167.3 km | Temps : 80 min | Coût : -2.5 $
    -> Borne_Kingston_Historic | Distance : 290 km | Temps : 140 min | Coût : 13 $
    -> Borne_Gatineau_Federal | Distance : 220 km | Temps : 90 min | Coût : 6 $
Borne : Borne_TroisRivieres_Principal
    -> Borne_Montreal_Ouest | Distance : 122.6 km | Temps : 70 min | Coût : 9.03 $
Borne : Borne_Windsor_Frontiere
    (aucun trajet sortant)
Borne : Borne_Ottawa_Capitale
    -> Borne_Toronto_Metro | Distance : 450 km | Temps : 250 min | Coût : 24 $
Borne : Borne_Quebec_Centre
    -> Borne_TroisRivieres_Principal | Distance : 114.6 km | Temps : 72 min | Coût : -5 $
    -> Borne_Sherbrooke_Universite | Distance : 150 km | Temps : 40 min | Coût : 12 $
    -> Borne_Chicoutimi_Regional | Distance : 210 km | Temps : 85 min | Coût : 3 $
==========================================

+================================================+
|                MENU PRINCIPAL                  |
+================================================+
```

```
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Quebec_Centre
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Quebec_Centre -> Borne_TroisRivieres_Principal ->
    Borne_Montreal_Ouest
Coût total pour ce critère : 237.2 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Quebec_Centre
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : temps de recharge
Chemin trouvé : Borne_Quebec_Centre -> Borne_Sherbrooke_Universite ->
   Borne_Montreal_Ouest
Coût total pour ce critère : 81 min
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
```

```
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Sherbrooke_Universite
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Sherbrooke_Universite -> Borne_Montreal_Ouest
Coût total pour ce critère : 155 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Quebec_Centre
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Quebec_Centre -> Borne_TroisRivieres_Principal ->
    Borne_Montreal_Ouest
Coût total pour ce critère : 237.2 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Quebec_Centre
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
```

```
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : temps de recharge
Chemin trouvé : Borne_Quebec_Centre -> Borne_Sherbrooke_Universite ->
   Borne_Montreal_Ouest
Coût total pour ce critère : 81 min
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Quebec_Centre
Nom de la borne de destination :   > Saisie : Borne_Montreal_Ouest
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : coût énergétique
Chemin trouvé : Borne_Quebec_Centre -> Borne_Chicoutimi_Regional ->
   Borne_TroisRivieres_Principal -> Borne_Montreal_Ouest
Coût total pour ce critère : 2.03 $
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Toronto_Metro
Nom de la borne de destination :   > Saisie : Borne_Windsor_Frontiere
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : distance
```

```
Chemin trouvé : Borne_Toronto_Metro -> Borne_Hamilton_Steel ->
   Borne_London_Academic -> Borne_Windsor_Frontiere
Coût total pour ce critère : 388 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Toronto_Metro
Nom de la borne de destination :   > Saisie : Borne_Sudbury_Minier
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Toronto_Metro -> Borne_Sudbury_Minier
Coût total pour ce critère : 410 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Toronto_Metro
Nom de la borne de destination :   > Saisie : Borne_Sudbury_Minier
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : temps de recharge
Chemin trouvé : Borne_Toronto_Metro -> Borne_Sudbury_Minier
Coût total pour ce critère : 200 min
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
```

```
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Toronto_Metro
Nom de la borne de destination :   > Saisie : Borne_Kingston_Historic
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Toronto_Metro -> Borne_Kingston_Historic
Coût total pour ce critère : 260 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Toronto_Metro
Nom de la borne de destination :   > Saisie : Borne_Kingston_Historic
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : temps de recharge
Chemin trouvé : Borne_Toronto_Metro -> Borne_Kingston_Historic
Coût total pour ce critère : 120 min
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
```

```
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Gatineau_Federal
Nom de la borne de destination :   > Saisie : Borne_Toronto_Metro
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : coût énergétique
Chemin trouvé : Borne_Gatineau_Federal -> Borne_Toronto_Metro
Coût total pour ce critère : 4 $
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 3
[RECHERCHE DIJKSTRA]
Nom de la borne d'origine :   > Saisie : Borne_Gatineau_Federal
Nom de la borne de destination :   > Saisie : Borne_Toronto_Metro
Critère de recherche :
  1. Distance
  2. Temps de recharge
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Gatineau_Federal -> Borne_Toronto_Metro
Coût total pour ce critère : 460 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_London_Academic
Nom de la borne de destination :   > Saisie : Borne_Toronto_Metro
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
```

```
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_London_Academic -> Borne_Toronto_Metro
Coût total pour ce critère : 205 km
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_London_Academic
Nom de la borne de destination :   > Saisie : Borne_Toronto_Metro
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : temps de recharge
Chemin trouvé : Borne_London_Academic -> Borne_Toronto_Metro
Coût total pour ce critère : 110 min
+================================================+
|                MENU PRINCIPAL                  |
+================================================+
| 1. Charger le réseau depuis un fichier         |
| 2. Afficher le réseau                          |
| 3. Rechercher un plus court chemin (Dijkstra)  |
| 4. Rechercher un plus court chemin (Bellman)   |
| 0. Quitter                                     |
+================================================+
Votre choix :
Option choisie : 4
[RECHERCHE BELLMAN-FORD]
Nom de la borne d'origine :   > Saisie : Borne_Sudbury_Minier
Nom de la borne de destination :   > Saisie : Borne_Toronto_Metro
Critère de recherche :
  1. Distance
  2. Temps de recharge
  3. Coût énergétique
Votre choix :   > Critère choisi : distance
Chemin trouvé : Borne_Sudbury_Minier -> Borne_Toronto_Metro
Coût total pour ce critère : 410 km
```

10 Fichiers à remettre

Vous devez rendre un fichier .zip comportant uniquement les fichiers suivants :

- L'ensemble des fichiers .h, et .cpp (il est important d'ajouter les balises Doxygen dans les fichiers .cpp, votre nom dans la section author ainsi que des commentaires à l'intérieur des méthodes pour expliquer votre méthode d'implémentation et ainsi faciliter le travail de correction).
- Le fichier contenant le barème de correction **NoteTp2-IFT2008.xlsx** (il faut juste ajouter votre nom et matricule sur la première ligne ainsi qu'éventuellement la version C++ utilisée et il ne faut bien sûr pas modifier le barème ou les formules Excel utilisées).

Vous ne devez en aucun cas ajouter un autre fichier ou répertoire. N'incluez aucun exécutable, ni aucun fichier généré par votre environnement de développement **(surtout pas le répertoire debug S.V.P.)**. De plus, vous n'avez pas à générer la documentation Doxygen en format html pour l'ensemble du TP (pour ne pas envoyer un gros fichier zip). Le correcteur pourra le faire lors de la correction. Vous devez également vérifier que vous n'avez pas inclus des fichiers ou répertoires cachés (i.e ___MACOSX, .git, .ide, .DS_Store, etc.). Le nom du .zip doit respecter le format suivant : TP2-Matricule.zip. Nous vous rappelons qu'il est important de faire la remise par voie électronique uniquement en vous connectant à : http://monportail.ulaval.ca/ (aucun travail envoyé par courriel n'est accepté). Il est toujours possible de faire plusieurs remises pour le même travail. Vous pouvez donc remettre votre travail plus tôt afin d'être sûr qu'il a été remis en temps, et remettre par la suite une version corrigée avant l'heure limite de remise. De plus, il est de votre responsabilité de vérifier après la remise que vous nous avez envoyé les bons fichiers (non vides et non corrompus), sinon vous pouvez avoir un zéro.

Attention ! Il est **formellement interdit de partager ou publier** le code du TP ou votre solution sur le Web avant ou après la remise du TP, car c'est une source évidente de plagiat et vous risquez donc d'être pénalisé. De plus, **tout travail remis en retard se verra pénalisé de -25% de la note.** Le retard débute dès la limite de remise dépassée (dès la première minute). Un retard excédant une journée provoquera le rejet du travail pour la correction et la note de 0 pour ce travail. De plus, tout travail doit être original. **Il n'est donc pas autorisé d'utiliser l'IA générative pour produire du code.** Toute ressemblance entre les copies sera signalée.

11 Remarques importantes

1. **Normes de programmation :** Vous devez respecter les normes de programmation du cours surtout en faisant correctement l'indentation du code. Vous devez également utiliser les balises Doxygen pour commenter votre code.
2. **Tolérance zéro vis-à-vis des erreurs de compilation :** Vos programmes doivent contenir zéro erreur de syntaxe. Un programme qui ne compile pas, peu importe la raison, se verra attribuer la note zéro.
3. **Portabilité des programmes :** Vos programmes doivent absolument pouvoir être compilés et exécutés sans erreurs et sans warnings. Autrement, vous pouvez avoir la note zéro pour votre travail. Testez intensivement votre programme. Pour cela, nous recommandons l'utilisation de Google Test. **Par contre, SVP, ne pas remettre vos testeurs**. Rappelez-vous qu'un programme qui ne plante pas n'est pas nécessairement sans bogues, mais qu'un programme qui plante même une seule fois comporte nécessairement un bogue. Si vous ne

testez pas suffisamment votre travail, il risque de provoquer des erreurs à l'exécution lors de la correction. La moindre erreur d'exécution rend la correction extrêmement difficile et vous en serez donc fortement pénalisés.

4. **Plagiat :** Ce travail doit être de votre propre création. Tout étudiant qui commet une infraction au Règlement disciplinaire à l'intention des étudiants de l'Université Laval dans le cadre du présent cours, notamment en matière de plagiat, est passible des sanctions qui sont prévues dans ce règlement. Il est très important pour tout étudiant de prendre connaissance des articles 23 à 46 du Règlement disciplinaire. Celui-ci peut être consulté à l'adresse suivante : http://ulaval.ca/reglement-disciplinaire

   Tout étudiant est tenu de respecter les règles relatives au plagiat. De plus, les étudiants ou étudiantes ayant collaboré au plagiat seront soumis aux sanctions normalement prévues à cet effet par les règlements de l'Université. Tous les travaux sont soumis à un logiciel de détection de plagiat.

5. **Pourquoi l'utilisation de l'IA générative est interdite :** Les grands modèles de langues (LLMs, de l'anglais Large Language Models) permettent de dialoguer dans plusieurs langages, incluant certains langages de programmation. Ces LLMs (comme ChatGPT) permettent d'accélérer la tâche de programmation, mais ne remplacent pas les experts. Les LLMs sont eux-mêmes des programmes informatiques, les informaticiens sont les mieux placés pour les comprendre et les améliorer. Il faut connaître les bases de la programmation pour vérifier le code généré par un LLM. Il faut avoir déjà programmé soi-même pour demander précisément ce que l'on veut à un LLM.

   Un des objectifs principaux de ce cours est de vous amener à développer des compétences pour développer des programmes en C++ et non à développer des compétences à bien rédiger des prompts. Les travaux pratiques ont pour objectif de vous permettre de valider votre compréhension de la matière enseignée avant l'examen. C'est donc dans l'intérêt des étudiants de compter sur eux même surtout qu'ils n'auront pas ces LLMs le jour de l'examen. D'ailleurs, l'intelligence artificielle générative ne peut pas vous dire :

   - Comment résoudre des bogues dans le code.
   - Comment intégrer le code dans un système.
   - Est-ce que le code est optimal.
   - Est-ce que le code est robuste.
   - Est-ce que le code est sécure.
   - Est-ce que le code respecte les bonnes pratiques.

Bon travail !