Automne 2015

Programmation avancée en C++

GIF-1003

Thierry EUDE, Ph.D

Travail individuel
à rendre avant le
jeudi 15 octobre 12h00 (midi)
(Voir modalités de remise à la fin de l'énoncé)

Tout travail remis constitue une contribution originale et distincte de travaux remis par d'autres. Le plagiat est strictement défendu. Tout travail plagié obtiendra la note 0. 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.

Par ailleurs, vu l'importance des communications écrites dans le domaine de la programmation, il sera tenu compte autant de la présentation que de la qualité du français et ce, dans une limite de 10% des points accordés.

Travail Pratique 2
Développement de classes

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

Notre objectif final est de construire un outil permettant au service de l'aménagement du territoire de la ville de Québec de localiser rapidement les bornes de paiement (horodateurs) et de stationnement. La série des travaux pratiques devrait tendre vers cet objectif. Chaque travail pratique constituera donc une des étapes de la construction de cet outil.

But du travail

- Apprivoiser le processus de développement d'une classe dans un environnement d'implémentation structuré,
- Respecter des normes de programmation et de documentation.

Classe Borne

Cette classe permet de modéliser tous les types de bornes présents dans la ville de Québec. Vous devez implanter les spécifications qui suivent dans les fichiers `Borne.h` et `Borne.cpp`.

Attributs de la classe

`std::string m_numBorne;`
Le numéro de la borne. Il doit être non vide.

`std::string m_coteRue;`
Le coté par rapport au centre de chaussée où est la borne. Il doit être non vide.

`double m_lectureMetrique;`
La distance mesurée à partir du début du tronçon dans le sens des numéros d'immeuble.

`int m_segmentRue;`
L'identifiant du segment de voie publique.

`std::string m_direction;`
Le coté du centre de chaussée ou l'intersection dans le cas d'un terre-plein.

`std::string m_nomTopographique;`
Le nom topographique (générique, liaison, spécifique, direction) du centre de chaussée.

Méthodes de la classe

Constructeur de la classe. On construit un objet `Borne` à partir de données passées en paramètre.

```
Borne (   const std::string& p_numBorne,
          const std::string& p_coteRue,
          double p_lectureMetrique,
          int p_segmentRue,
          const std::string& p_direction,
          const std::string& p_nomTopographique);
```

Méthodes d'accès aux attributs de la classe :

```
reqNumBorne
reqCoteRue
reqLectureMetrique
reqSegmentRue
reqDirection
reqNomTopographique
```

`void asgNomTopographique(const std::string& p_nomTopographique)`
Méthode permettant de changer le nom topographique de la borne.

`std::string reqBorneFormate() const;`
Méthode retournant dans un objet `std::string` les informations correspondant à une borne formatées sous le format suivant :

```
Numero de la borne      : 2172
Cote de la rue    : Nord
Distance mesuree  : 23.7
Segment de rue    : 20
Direction         : Nord
Nom topographique : Boulevard René-Levesque Est
```

Utilisez la classe ostringstream de la bibliothèque standard pour formater les informations sur la borne.

Opérateur de comparaison d'égalité. La comparaison se fait sur la base de tous les attributs.
`bool operator==(const Borne& p_borne)`

Documentation

La classe `Borne` ainsi que toutes les méthodes devront être correctement documentées pour pouvoir générer une documentation complète à l'aide de l'extracteur DOXYGEN. Des précisions sont fournies sur le site Web pour vous permettre de l'utiliser (syntaxe et balises à respecter etc.).
Vous devez respecter les normes de programmation adoptée pour le cours. Ces normes de programmation sont à votre disposition dans la section "Notes de cours / Présentations utilisées en cours / Normes de programmation en C++ " du site Web du cours.
Aussi, comme indiqué dans ces normes, vous devez définir un espace de nom (namespace) qui inclura le code spécifique au développement de l'outil d'aide pour la localisation des bornes (pour le moment il n'y aura que la classe Borne). Il devra porter le nom suivant : `tp` (en minuscules).

Utilisation

Après avoir implanté la classe comme demandé, vous devez écrire un programme interactif minimaliste. Le but est simple, il s'agit simplement d'obtenir interactivement avec l'usager, les données nécessaires pour créer une "*`Borne`*" puis de modifier son adresse.
Rappelons qu'un objet *`Borne`* ne peut être construit qu'avec des valeurs valides. C'est la responsabilité du programme principal qui l'utilise de s'assurer que ces valeurs sont valides.
Les critères de validité ont été énoncés dans la description des attributs des classes.
Une fois toutes les données validées, vous créez un objet *`Borne`* et vous demandez à l'objet créé de retourner une chaîne formatée (*`reqBorneFormate()`*) pour pouvoir l'afficher dans la console.

Modalités de remise, bien livrable

Le deuxième travail pratique pour le cours GIF-1003 Programmation avancée en C++ est un **travail individuel**. Vous devez remettre votre environnement de développement, soit le code source dans un espace de travail (workspace) Eclipse mis dans une archive en utilisant le dépôt de l'ENA : https://www.portaildescours.ulaval.ca/ena/site/evaluation?idSite=64498&idEvaluation=221004&onglet=boiteDepots
Ce travail est intitulé TP #2. **Aucune remise par courriel n'est acceptée**.
Vous pouvez remettre autant de versions que vous le désirez. Seul le dernier dépôt est conservé. **Il est de votre responsabilité de vous assurer de ce que vous avez déposé sur le serveur**.

Attention, vérifiez qu'une fois déplacé et décompressé, votre workspace est toujours fonctionnel (il doit donc être « portable »). Pensez au correcteur ! Sachez qu'il utilisera la machine virtuelle fournie pour le cours.

Très important

Ne pas utiliser « windows » pour faire votre archive ; son outil natif présente des lacunes. Utilisez à la place 7-zip (gratuit, inclus dans la machine virtuelle fournie pour le cours), ou winrar (shareware), ou mieux, faites-le dans la machine virtuelle du cours.

Date de remise

Ce travail doit être rendu avant le **jeudi 15 octobre 2014 12h00 (midi)**. Pour tout retard non motivé (voir plan de cours; motifs acceptables pour s'absenter à un examen), la note 0 sera attribuée.

Critères d'évaluation

1) Respect des normes de programmation, et de documentation
2) Structures, organisation du code (très important!)
3) Exactitude du code
4) Utilisation des classes, i.e. le programme principal

Particularités du barème

- *Si des pénalités sont appliquées, elles le sont sur l'ensemble des points.*
- *Si un travail comporte ne serait-ce qu'une erreur de compilation, il sera fortement pénalisé, et peut même se voir attribuer la note zéro systématiquement.*
- *Il est très important que votre travail respecte strictement les consignes indiquées dans l'énoncé, en particulier les noms des méthodes, les noms des fichiers et la structure de développement sous Eclipse sous peine de fortes pénalités*

Bon travail