Automne 2021

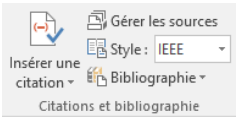
Programmation avancée en C++

GIF-1003

Thierry EUDE, Ph.D

Travail individuel

à rendre avant
jeudi 21 octobre 2021 14h
(voir modalités de remise à la fin de l'énoncé)



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. Constitue notamment du plagiat le fait de:

- remettre un travail copié d'un autre étudiant (avec ou sans l'accord de cet autre étudiant);
- remettre un travail téléchargé d'un site d'achat ou d'échange de travaux scolaires.

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.

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 de gestion de références bibliographiques. La série de 4 travaux pratiques devrait tendre vers cet objectif. Chaque travail pratique constituera donc une des étapes de la construction de cet outil.

But du deuxième travail

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

Classe Reference

Cette classe permet de modéliser des références bibliographiques. Vous devez implanter les spécifications qui suivent dans les fichiers `Reference.h` et `Reference.cpp`.

Attributs de la classe

`m_auteurs`
Un objet string; le nom l'auteur (ou du premier auteur s'ils sont plusieurs) de la référence. Il doit être non vide et doit être dans un format valide tel que déterminé par la fonction déjà développée (TP1)

```
bool validerFormatNom (const std::string& p_nom)
```

`m_titre`
Un objet string; le titre doit être non vide et peut comporter des espaces.

`m_annee`
Un entier; l'année d'édition de la référence. Elle doit être strictement plus grande que 0.

`m_identifiant`
Un objet string; l'identifiant de la référence. Il peut être un code ISSN ou ISBN.
S'il s'agit d'un code ISSN, il doit être dans un format valide tel que déterminé par la fonction déjà développée `bool validerCodeIssn (const std::string& p_issn)`. Cette fonction valide le code ISSN d'une publication en série (voir TP1).
S'il s'agit d'un code ISBN, il doit être dans un format valide tel que déterminé par la fonction `bool validerCodeIsbn (const std::string& p_isbn)`. Cette fonction valide le code ISBN d'un ouvrage (livre) (voir TP1).

Méthodes de la classe

On vous demande de développer
Un constructeur de la classe. On construit un objet Reference à partir de données passées en paramètre.
Les données passées sont : l'auteur, le titre, l'année d'édition et l'identifiant.
Des méthodes d'accès en lecture aux attributs de la classe (accesseurs), ceci pour tous les attributs.
Une méthode permettant de modifier l'année d'édition de la référence en assignant une nouvel année d'édition à la référence courante (un mutateur). La nouvelle année doit bien entendu être valide tel que décrite plus haut pour l'année d'édition.

Une méthode `reqReferenceFormate` retournant dans un objet std::string les informations correspondant à une référence formatées sous le format suivant :
`Homayoon Beigi, Fundamentals of Speaker Recognition, 2011, ISBN 978-0-387-77591-3.`
Utilisez la classe ostringstream de la librairie standard pour formater les informations.

Un opérateur de comparaison d'égalité. La comparaison se fait sur la base de tous les attributs.

Fonction de validation

Les fonctions de validation ont été développées au TP1 dans le module `validationFormat`. Il s'agit ici de les utiliser pour vérifier les données avant de les transmettre aux méthodes. Insérez ces fonctions dans le namespace `util`. Les modifications apportées au code initial du TP1, peuvent être qualifiées de maintenance.

Documentation

La classe `Reference` 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). Il devra porter le nom suivant : `biblio` (en minuscules).

Utilisation `(gestionReferences.cpp)`

Après avoir implanté et testé la classe `Reference`, 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 "Référence", l'éditer à l'écran (affichage des informations formatées), proposer de modifier l'année d'édition, l'éditer à nouveau pour pouvoir constater que la modification est effective.

Rappelons qu'une référence ne devrait être construite 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 de la classe `Reference`.

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 le code source dans un projet Netbeans mis dans une archive 7z (voir le tutoriel sur la page des travaux pratiques dans contenu et activités), en utilisant le dépôt de l'ENA (mon portail).

Ce travail est intitulé TP2. Aucune remise par courriel n'est acceptée.

Très important

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

Rappel: un tutoriel est disponible sur la page des travaux pratiques pour vous guider.
Vous pouvez remettre autant de versions que vous le désirez.
Pensez à supprimer vos anciennes versions sur l'ENA pour ne laisser que celle qui sera corrigée. **Il est de votre responsabilité de vous assurer de ce que vous avez déposé sur le serveur.**

Date de remise

Ce travail doit être rendu avant le **jeudi 21 octobre 2021 14h00**. 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 biens livrables
2) Respect des normes de programmation, et de documentation
3) Structures, organisation du code (très important!)
4) Exactitude du code
5) Utilisation des classes, c.-à-d. 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 Netbeans sous peine de fortes pénalités*

Bon travail