ÉCOLE POLYTECH'NICE SOPHIA
SI4
2016-2017

Julien Deantoni

Programmation orientée objet : CPP

TD numéro 4

Expressions...

Objectif

L'objectif de cet exercice est de comprendre du code existant et de le modifier dans le but de refléter un changement de spécification. Le code partiel correspondant à une spécification numéro 1 est fourni sur mon site web. Il s'agira dans un premier temps de rajouter une fonctionnalitée à cette spécification. Ensuite, vous passerez à la modification du code pour qu'il corresponde à une deuxième spécification.
Pensez à utiliser l'utilitaire `valgrind` pour vérifier que vous n'avez pas de fuites mémoires.

Spécification numéro 1

Le code fourni correspond à une partie de l'implémentation des classes permettant de manipuler des expressions. La spécification est donnée figure 1. Remarquez que les expressions unaires et binaires ne contiennent pas leur(s) opérande(s).
Vous devez ajouter la fonction eval(), qui renvoie un entier résultat de l'expression.
Ce code n'est pas le plus propre qui puisse exister. Entre autres, il demande à l'utilisateur de manipuler des pointeurs dans les arguments des constructeurs. Modifier les sources fournies pour éviter ce problème. Réfléchissez et demandez vous s'il est possible de n'utilisez aucun pointeur. Pensez aux éventuels problèmes / restrictions, que vous consignerez dans votre code source en tant que commentaires. Si vous pensez que c'est possible, faites le.
Remarquez bien que vos modification ne doivent en aucun cas modifiez la specification partielle fournie par la figure 1.

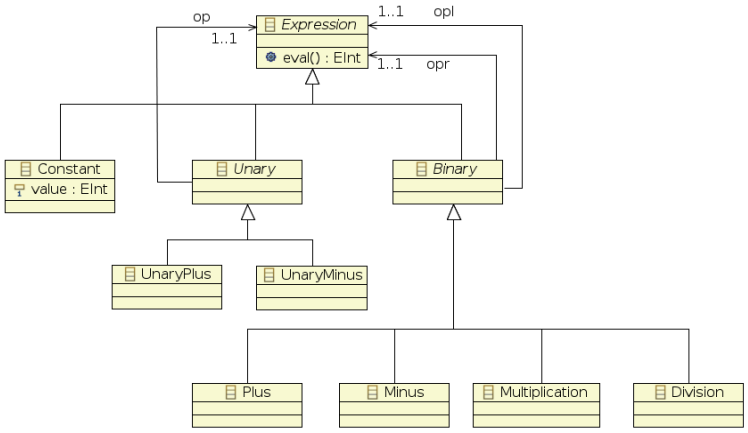

Figure 1: Spécification numéro 1

Spécification numéro 2

On vous demande de modifier le code précédent afin que les expressions unaires et binaires contiennent leur(s) opérande(s) (confère Figure = 2).
Pensez à vous assurer que vous n'avez pas de fuites mémoires.

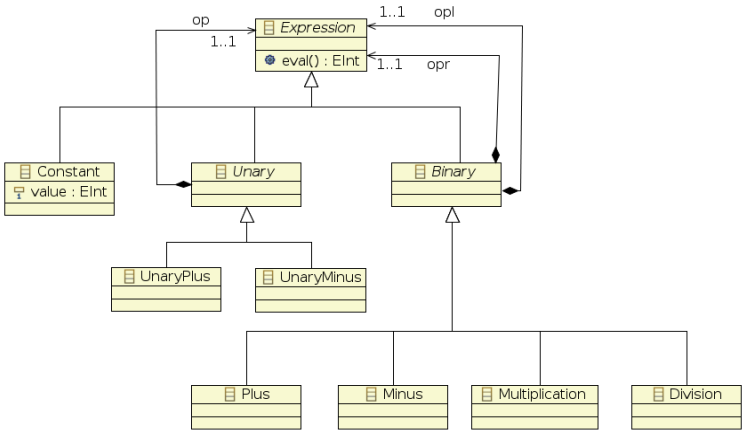

Figure 2: Spécification numéro 2

Suppléments

On propose de réaliser une extension qui consiste à ajouter à la hiérarchie des nœuds d'expression deux nouveaux opérateurs :

1. l'opérateur (binaire) de modulo (opérateur % en C),
2. l'opérateur (ternaire) conditionnel ("`if then else`").