Travaux dirigés C++ n°2

Informatique

—IMAC 2e année—

Introduction à l'Objet

Ce TD a pour but d'introduire le concept d'objet en C++. Les étudiants apprendront à écrire une classe, son constructeur et son destructeur.

▸ Exercice 1. *Une classe*

Objectif de l'exercice : Créer une première classe. Révisions : class

A faire

Vous devez créer une classe de vecteur de double à n dimensions. Votre programme contiendra la fonction main habituelle dans un fichier main.cpp, ainsi que la classe VectorD située dans deux fichiers VectorD.hpp et VectorD.cpp. Les fichiers doivent respecter les standards déjà définis. Instanciez un objet VectorD dans la fonction main.

Questions

- *Quels peuvent être les attributs et les méthodes de cette classe ?*
- *Cette classe a t'elle besoin de paramètres à la création? Si oui, lesquels?*

▸ Exercice 2. *Construisons*

Objectif de l'exercice : Comprendre le mécanisme de construction. Révisions : Constructeur, new

Questions

- *Dans quel cas est créé le constructeur par défaut?*
- *Rappelez la signification du mot clé this.*

A faire

Si ce n'est pas encore fait, dotez votre vecteur d'un tableau de double alloué dynamiquement. Ce tableau doit être alloué à la construction de l'objet. Écrivez le constructeur qui permet d'implémenter cette fonctionalité.

▸ **Exercice 3.** ***Recopions***

Objectif de l'exercice : Savoir implémenter un constructeur par recopie Révisions : Constructeur par recopie, références

Questions

- *Si vous ne créez pas de constructeur par recopie, le code suivant est il valide ?*

```
VectorD b(10); VectorD a = b;
```

- *Que se passe t'il alors?*

A faire *Implémentez et testez le constructeur par recopie de la classe VectorD. Affichez une chaîne de caractère différente pour chaque constructeur et observez lequel est appelé pour chaque cas.*

▸ **Exercice 4.** ***Détruisons***

Objectif de l'exercice : Maîtriser le mécanisme du destructeur Révisions : destructeur, delete[]

- *Testez votre programme actuel en utilisant valgrind qu'observez vous?*

A faire *Écrivez le destructeur pour VectorD qui permettra de résoudre le problème rencontré. Testez à nouveau avec valgrind.*

▸ **Exercice 5.** ***Protection et encapsulation***

Objectif de l'exercice : Implémenter le concept d'encapsulation et de protection des données. Révisions : public, private

Questions

- *Rappelez la signification des mots clés public et private.*
- *Expliquez l'intérêt de protéger les membres et fonctions d'une classe.*
- *A quoi sert le mot-clé inline*

A faire *Vous devez reprendre la classe VectorD précédemment créée et faire en sorte que les membres de cette classes deviennent privés. Vous créerez donc les accesseurs en lecture et en écriture au besoin. Pensez à utiliser les mot clés const et inline lorsqu'ils sont appropriés.*

▸ **Exercice 6.** ***Fonctionnons***

Objectif de l'exercice : *Implementer des fonctions au sein d'une classe.* *Révisions :* *déclarations de fonctions*

A faire *Votre vecteur n'est pour l'instant pas très utile. Ajoutez lui la capacité de se normaliser, donnez la possibilité de lui additionner un autre vecteur. Reflechissez bien aux paramètres et valeurs de retour de ces fonctions.*

▸ **Exercice 7.** ***Serialisons*** ⋆

Objectif de l'exercice : *Réviser les entrées/sorties fichier.* *Révisions :* *i/o binaires*

A faire *Faites en sorte de pouvoir sauvegarder un tableau de vos vecteurs dans un fichier à l'aide de* `fstream`*. Vous ferez ensuite en sorte de pouvoir reconstruire ce tableau à partir du même fichier.*