ISIMA 3ème année - MODL/C++
TP 2 : Généricité

Pré-requis

On reprend le premier exercice du TP précédent. On suppose que les classes **Point**, **Cartésien** et **Polaire** ont été correctement définies et sont fonctionnelles, et que les moyens de conversion entre **Cartésien** et **Polaire** existent. Copiez le code source de ces classes dans le répertoire tp_2/src et modifiez CMakeLists.txt pour que CMake les prenne en compte.

Ce TP vous fera également découvrir un autre outil de génie logiciel : **Git**. Il s'agit d'un **système de gestion de version** permettant notamment de conserver un historique des modifications d'un projet à travers l'identification de versions, et donc de pouvoir revenir en arrière si nécessaire. Il permet également un travail collaboratif sur un projet. Un document introductif à Git vous est fourni : il vous explique les principes et les commandes principales de l'outil.

Exercice

Il vous est demandé d'utiliser l'outil Git pour accompagner le développement de cet exercice. Vous devez donc créer un dépôt (*repository*) en local sur votre compte, et valider une nouvelle version à chaque fin de question (*commit*).

1) Définir la classe **Nuage**, générique sur le type de points à contenir. Utiliser la classe std::vector pour stocker les points. Cette classe devra fournir des itérateurs, à la manière STL (méthodes begin() et end(), type iterator). Tests 1-2

2) Définir une **fonction** générique barycentre_v1() prenant en argument un nuage de points et retournant le barycentre, supposé du même type que les points. Proposer une solution générique qui s'appuie sur la formule connue pour les cartésiens (utiliser des conversions). Tests 3-4a

3) Cette première version générique nécessite la conversion systématique en cartésien de tous les points, ce qui a un coût lorsqu'on manipule des polaires. En supposant qu'on dispose de la formule du barycentre en coordonnées polaires, proposer une spécialisation pour les polaires de la fonction générique barycentre_v1(). Test 4b

   Nous n'avons pas cette formule, donc faire à la place une moyenne des angles et des distances (mais attention, le test 4a devient invalide, il faut le désactiver).

Dans la prochaine question, vous devez proposer une solution alternative à celle des questions précédentes, il vous est donc demandé de créer une nouvelle branche dans votre dépôt, afin de bien identifier les deux possibilités.

4) On souhaite que la fonction de barycentre s'affranchisse aussi du type du conteneur de points. Pour cela, la classe **Nuage** propose une interface similaire aux conteneurs STL, à savoir des méthodes begin() et end(), ainsi qu'un type interne iterator. Proposer une fonction barycentre_v2() qui puisse fonctionner indifféremment sur les principaux conteneurs STL et la classe **Nuage**. Tests 5-7