TD3 : Classe et objet (1/2)

Exercice 1 : Conception et spécificateur

On souhaite créer un type de donnée pour représenter une personne. Cette personne est identifiée par son nom, son prénom et son numéro de sécurité sociale (numéro unique). On veut pouvoir saisir et afficher ces informations.

a. Ecrivez, en C++, la classe Personne.

b. Donner le programme principal, en C++, permettant de saisir puis d'afficher les informations d'une personne.

c. En programmant la procédure membre d'affichage, vous avez dû choisir sous quel format la personne est affichée, et en particulier quel(s) caractère(s) de séparation utiliser entre les différentes données membres à afficher. Ecrire une deuxième version de cette procédure, utilisant le principe de surcharge, permettant de paramétrer le(s) caractère(s) utilisé(s).

d. Modifier cette procédure pour qu'elle prenne une valeur de paramètre par défaut. Voyez-vous un problème ?

e. Que doit-on faire pour interdire aux utilisateurs de la classe de manipuler les données membres directement ?

f. Afin de permettre leur manipulation, écrivez des procédures/fonctions dont le rôle est de lire et modifier les données membres. Quels sont les avantages et inconvénients de cette conception ?

g. Ecrivez un programme principal qui modifie le nom d'une personne après saisie.

Exercice 2 : Surcharge d'opérateurs

On souhaite pouvoir faire des opérations mathématiques sur des points 2D. Une solution serait de définir des fonctions membres telles que `void additionnerPoints(const Point2D & p)`. Une autre solution consiste à définir les opérateurs élémentaires pour un point 2D de telle façon à ce que l'on puisse exécuter des instructions telles que `pt3=pt1+pt2;`.

a. Ecrire la classe Point2D avec les fonctionnalités nécessaires pour additioner, soustraire et tester l'égalité de deux points.

b. Ajouter les fonctionnalités d'affectation et d'incrément préfixé (ajout de 1 à chaque coordonnée lorsque l'on fait `++pt`).

c. Ajouter les fonctionnalités de symétrie centrale par rapport à l'origine définie par l'opérateur unaire « - », et le test si le point est situé à l'origine avec l'opérateur « ! ».

d. Ajouter les fonctionnalités d'affichage et de lecture des coordonnées du point suivant le format `( x , y )`.