TP-4 POO en C++ (4 ème Séance)

Ce TP aborde les points suivants :
- Constructeur de copie
- Fonctions Amies
- Surdéfinition d'opérateurs

Chapitre 3, 4, 5 du Polycopie : Classes (fonctions amies), Surcharge d'opérateurs

Exercice 1 : Surdéfinition d'opérateurs de la classe Complexe

On utilisera la classe « Complexe du TP-3 »
Q1.1 Définir les opérateurs « == », « = ! » et l'opérateur « ! »
Q1.2 Définir l'opérateur d'affectation « = »
Q1.3 Définir les opérateur « + », « - », « * »
Q1.4 surcharger les opérateurs << et >> pour que nous puissions afficher et saisir un complexe : (cout<<c1 ; cin>>c2)

Exercice 2 : Surdéfinition d'opérateurs de la classe vecteur3D

On utilisera la classe « Vecteur3d » crée dans le TP-3.
Q2.1 Définir les opérateurs « == et = ! » de manière à ce qu'ils permettent de tester la coïncidence ou le non-coïncidence de deux points (on utiliser des fonctions membres ou fonctions amies).
Q2.2 Définir les opérateurs + pour qu'ils fournissent la somme de deux vecteurs et l'opérateur * pour qu'il fournisse le produit scalaire de deux vecteurs (Il est préférable d'utiliser les fonctions amies).
Q2.3 surcharger les opérateurs << et >> pour que nous puissions afficher et saisir un Vecteur3d : (cout<<v1 ; cin>>v2)

Exercice 3 : Classe String

On désire concevoir une classe chaîne de caractères String de longueur quelconque.

1. Ecrire la déclaration de cette classe dans un fichier ***string1.h***.

Méthodes souhaitées :
- ***longueur()*** : retourne la longueur de la chaîne
- ***nieme()*** : retourne le n ième caractère
- ***affiche()*** : affiche la chaîne
- ***saisie()*** : saisie d'une chaîne d'au plus 1024 caractères
- ***concatene()*** : ajoute une chaîne ou un caractère à la fin de la chaîne
- ***egal()*** : retourne 1 si les chaînes sont égales
- ***minuscule()*** : retourne une nouvelle chaîne formée des éléments de la chaîne d'origine mis en minuscules.

2. Après avoir fait valider cette déclaration, vous devez entreprendre la définition des méthodes dans le fichier ***string1.cpp***.
3. Ensuite utiliser le programme essaiString1.cpp suivant pour valider la classe.

```
#include <iostream.h>
#include "string1.h"

void main() {
    String ch1("essai"), ch2 = ch1, ch3('=' , 80);
    const String ch4("chaîne constante");
    ch1.nieme(1) = 'E'; // le premier caractère de la chaîne
    cout << ch4.nieme(1) << endl;
    ch2.saisie();
    ch2.concatene(" de la classe String");
    ch2.concatene('g');
    if ( ! egal(ch2, "") ) {
     ch2.affiche();
     cout << endl;
    }
    ch2.minuscule().affiche();// est-ce bien raisonnable ???
    cout << endl;
}
```

Attention : des contraintes supplémentaires pourront être déduites du programme d'essai donné.