Programmation C++
Parcours MF– ING2
TP3 : Surcharge des opérateurs

Exercice 1 : Classe Date

Le point de départ du TP est la classe Date définie comme suite :

```
class Date {
        private :
                short jour;
                short mois;
                int annee;
        public :
                Date (short jour, short mois, int annee);
                ~Date();
};
```

L'objectif est de définir des fonctions de comparaison sur la classe Date en utilisant la possibilité de surcharger les opérateurs du C++. Par exemple, l'opérateur < peut être surchargé ainsi :

```
class Date {
        ...
        bool operator< (const Date &autre) const;
        // la date est-elle antérieure à l'autre date ?
        ...
};
```

Compléter la classe Date pour surcharger les 6 opérateurs de comparaison : ==, !=, <, >, <=, >=.

Expliquer comment utiliser les opérateurs de comparaison sur les dates ?

Écrire un programme de test qui permet de vérifier la classe Date ainsi écrite.

Il est également possible de surcharger un opérateur à l'extérieur de la classe. La signature est alors de la forme :

```
bool operator< (const Date &d1, const Date &d2)
// d1 est-elle antérieure à d2 ?
```

Que constatez-vous lorsque vous surchargez l'opérateur < de cette façon ? Pourquoi ?

Exercice 2 : Classe Vector (suite)

Compléter la classe Vector du TP1, avec le constructeur de copie, l'opérateur d'affection, …

Exercice 3 : classe sur les nombres complexes

Ecrire une classe qui permet de gérer des nombres complexes. Un nombre complexe possède une partie réelle et une partie imaginaire, et s'affiche de la manière suivante :

*"re + i * im"*. On doit pouvoir additionner, soustraire, multiplier, diviser et affecter des nombres complexes comme s'il s'agissait d'entiers ou de réels (c'est-à-dire, en surchargeant les opérateurs +, -, *, / et =). On doit également pouvoir créer un nombre complexe par recopie.

Ecrire un programme principal qui permet de tester cette classe.

Exercice 4 : Classe Duree

Le point de départ du TP est la classe Duree fournie non complète (à compléter). L'objectif est de définir des fonctions de comparaison sur la classe Duree. Ceci nous permettra de comprendre la différence entre une fonction membre, une fonction de classe et une fonction hors de classe ainsi que l'intérêt de pouvoir nommer le paramètre implicite (cas où *this* ne peut pas être omis).

4.1 : Surcharge des opérateurs

C++ offre la possibilité de surcharger les opérateurs du langage, en particulier les opérateurs de comparaison. Par exemple, l'opérateur < peut être surchargé ainsi :

```
class Duree {
...
bool operator< (const Duree &autre) const;
// la duree estelle antérieure à l'autre date ?
...
};
```

Compléter la classe Date pour surcharger les 6 opérateurs de comparaison.
Expliquer comment utiliser les opérateurs de comparaison sur les Dates ?
Écrire un programme de test qui permet de vérifier la classe Date ainsi écrite.

4.2 : Surcharge d'opérateurs hors de la classe

Il est également possible de surcharger un opérateur à l'extérieur de la classe. La signature est alors de la forme :

```
bool operator < (const Duree &d1, const Duree &d2);
// d1 estelle antérieure à d2 ?
```

Que constatez-vous lorsque vous surcharger l'opérateur < de cette façon ? Pourquoi ?