

## Laboratoire n°12 : patrons de fonction et de classe

Créer une classe date comprenant :

- les données membres privées de type entier : jour, mois, annee ;
- un constructeur initialisant les données membres ;
- une surcharge de l'opérateur << permettant d'afficher une date sous le format jour/mois/annee ;
- une surcharge de l'opérateur == permettant de comparer 2 dates, elle retourne true si les dates sont égales sinon false ;
- une surcharge de l'opérateur < qui retourne true si la première date est avant la deuxième date ;
- une surcharge de l'opérateur > qui retourne true si la première date est après la deuxième date.

Créer un patron de classe permettant de travailler sur des tableaux de types différents. Sa syntaxe de déclaration doit être la suivante :

```
template <class t, int taille>
```

```
class tableau {} ;
```

où t représente n'importe quel type et taille est la taille du tableau (lui donner une valeur par défaut).

Ce patron de classe comprend :

- une donnée membre de type tableau ;
- un constructeur initialisant la donnée membre ;
- une fonction affiche qui permet d'afficher tous les éléments du tableau ;
- une fonction tri qui permet de trier tous les éléments du tableau. ( tri a bulles)

Créer un programme qui :

- déclare un tableau d'entiers, un tableau de doubles, un tableau de caractères et un tableau de dates
- remplit ces tableaux en générant aléatoirement les éléments (fonction rand())
- crée un objet de type entier et de taille fixée, affiche ses éléments, les trie et réaffiche ses éléments
- crée un objet de type double et de taille fixée, affiche ses éléments, les trie et réaffiche ses éléments
- crée un objet de type char et de taille fixée, affiche ses éléments, les trie et réaffiche ses éléments
- crée un objet de type date et de taille fixée, affiche ses éléments, les trie et réaffiche ses éléments.

```

#include<iostream.h>
#include<string.h>
#include<stdlib.h>
#include<time.h>

class date{
    int jour,mois,annee;
public:
    date(int j=1,int m=1,int a=2006);
    friend ostream& operator<<(ostream& o,date d1);
    friend bool operator==(date d1,date d2);
    //friend bool operator<(date d1,date d2);
    friend bool operator>(date d1,date d2);
};
template <class t,int taille>
class tableau{
    t tab[taille];
public:
    tableau(t ta[]);
    void affiche();
    void tri();
};
date::date(int j,int m,int a){
    jour=j;
    mois=m;
    annee=a;
}
ostream& operator<<(ostream& o,date d1){
    o<<d1.jour<<"/"<<d1.mois<<"/"<<d1.annee<<endl;
    return o;
}
bool operator==(date d1,date d2){
    if((d1.jour==d2.jour)&&(d1.mois==d2.mois)&&(d1.annee==d2.annee)) return true;
    else return false;
}
/*
bool operator<(date d1,date d2){
    if(d1.annee<d2.annee){
        if(d1.mois<d2.mois){
            if(d1.jour<d2.jour){
                return true;
                //je n'utilise pas cette méthode car je
                n'utilise pas la surcharge de <
            }
            else return false;
        }
        else return false;
    }
    else return false;
}
*/

```

```

bool operator>(date d1,date d2){

return((d1.annee>d2.annee)||((d1.annee==d2.annee)&&((d1.mois>d2.mois)||((d1.mois==
d2.mois)&&(d1.jour>d2.jour)))));
}

template <class t,int taille>
tableau<t,taille>::tableau(t ta[]){
    for(int i=0;i<taille;i++){
        tab[i]=ta[i];
    }
}
template <class t,int taille>
void tableau<t,taille>::affiche(){
    for(int i=0;i<taille;i++)
        cout<<tab[i]<<" ";
    cout<<endl;
}
template <class t,int taille>
void tableau<t,taille>::tri(){

int max=taille-1;
t tmp;
for(int i=0;i<max-1;i++)
{
    for(int j=0;j<max-i;j++)
    {
        if(tab[j]>tab[j+1])
        {
            tmp=tab[j];
            tab[j]=tab[j+1];
            tab[j+1]=tmp;
        }
    }
}
}
}

```

```

void main(){

    srand(time(0));           //génère des éléments différents à chaque exécution
    int ti[10];
    for(int i=0;i<10;i++){
        ti[i]=int(rand()%50); //0 à 50
    }
    tableau<int,10> t1(ti);
    t1.affiche();
    t1.tri();
    cout<<"tableau trie:\n";
    t1.affiche();
    cout<<endl;

    double td[10];
    for(int j=0;j<10;j++){
        td[j]=(double)rand()/(((double)(68)+(double)(1))); //astuce pour générer des doubles
    }
    tableau<double,10> t2(td);
    t2.affiche();
    t2.tri();
    cout<<"tableau trie:\n";
    t2.affiche();
    cout<<endl;

    char tc[10];
    for(int k=0;k<10;k++){
        tc[k]=char(65+(rand()%26));
    }
    tableau<char,10> t3(tc);
    t3.affiche();
    t3.tri();
    cout<<"tableau trie:\n";
    t3.affiche();
    cout<<endl;

    date tdate[10];
    for(int l=0;l<10;l++){
        tdate[l]=date(((rand()%31)+1),((rand()%12)+1),((2000+(rand()%25))));
    }
    tableau<date,10> t4(tdate);
    t4.affiche();
    t4.tri();
    cout<<"tableau trie:\n";
    t4.affiche();
}

```