Fiche C++ n° 1

Principales notions abordées :
- **Classe et objet**
- **Notion de constructeur**
- **Membres privés et public**
- **Ecriture des fonctions à l'intérieur ou à l'extérieur de la classe**

Une classe correspond à une structure complexe dans laquelle vont se trouver des attributs (données membres) et des méthodes (fonctions).

1ère méthode : L'écriture des fonctions se fait à l'intérieur de la classe

Cette méthode est surtout utilisée dans le cas de fonctions très simples. Dans ce cas, on dit que les fonctions sont **inline.**

Le mot clé **class** permet de regrouper dans un même ensemble des données et des fonctions

Le mot clé **private** signifie que seuls les fonctions membres de la classe (c'est à dire les fonctions affiche(), deplace() et point()) pourront utiliser les attributs ou les méthodes qui suivent

Le mot clé **public** signifie que les attributs et opérations qui suivent peuvent être utilisées à la fois par les méthodes de la classe ou par des fonctions extérieures, en particulier la fonction principale

```
// fiche 1_1.cpp
#include <iostream>

using namespace std;

class Point
{
private:
        float y ;
        float x ;

public:
        Point(float x1=0 , float y1=0)
          {
          x=x1;
          y=y1;
          }

        void affiche(void)
          {
          cout <<"\n x="<<x<<"y="<<y;
          }


        void deplace(float tx, float ty)
          {
          x += tx;
          y += ty;
          }
};



//**** programme principal ***

int main(int argc, char* argv[])
{
        Point p1(2,4);
        p1.deplace(2,2);
        p1.affiche();
        return 0;
}
```

Cette fonction (méthode) porte le même nom que la classe. Il s'agit du constructeur. Cette fonction est appelée automatiquement quand on crée un objet de cette classe (ou une instance). Si lors de la création, aucun paramètre n'est indiqué, l'abscisse et l'ordonnée du point vaudront 0 (paramètres par défaut).
Le rôle essentiel des constructeurs est d'initialiser les données membres de la classe

Ne pas oublier le **;** à la fin de la définition de la classe

On appelle la fonction deplace() qui est une fonction de la classe point de la même manière que l'on accède à un élément d'une structure. Les paramètres 2 et 2 correspondent à la translation demandée en x et en y

Définition d'un objet p1 (instance de la classe point). Le simple fait de déclarer cet objet fait appel automatiquement au constructeur (fonction point()) qui a pour rôle d'initialiser l'abscisse de p1 à 2 et son ordonnée à 4.

Exercice : Que se produit-il si l'on essaie de modifier l'abscisse de p1 ? Pourquoi ? Résoudre le problème.

2ème méthode : L'écriture des fonctions se fait à l'extérieur de la classe

Cette façon de procéder est la plus courante. En effet, la plupart des classes comportent un nombre important de fonctions membres. Il serait donc lourd de toutes les écrire en même temps que l'on déclare la classe.

Ce fichier .h comporte les déclarations des données et fonctions membres de la classe

Les membres public et privés peuvent être placés dans n'importe quel ordre. Ce sont les mots clés public ou private qui indiquent le type des données ou fonctions qui suivent.

Si rien n'est indiqué, les données ou fonctions membres sont **privés par défaut**.

Si on avait utilisé le mot clé **struct** à la place du mot clé **class** les données ou fonctions membres auraient été **public par défaut**

```
// point.h (Déclarations)

class Point
{
public:
        void affiche(void);
        void deplace(float tx, float ty);

        Point(float x1=0 , float y1=0);

private:
        float y;
        float x;
};
```

Ne pas oublier le **;** à la fin de la déclaration des fonctions membres

```
// point.cpp: implementation

#include "point.h"
#include <iostream>

using namespace std ;

Point:: Point(float x1 , float y1)
{
x=x1;
y=y1;
}

void Point::deplace(float tx, float ty)
{
x+=tx;y+=ty;
}

void Point::affiche()
{
cout <<"\nx="<<x<<"y="<<y;
}
```

Si vous écrivez les fonctions membres d'une classe en dehors de cette classe, vous devez l'implémenter par la syntaxe :

TypeRetour NomClasse : : NomFonction(Liste Arguments)

```
// fiche 1_3.cpp
//**** programme principal ***

#include "point.h"

int main(int argc, char* argv[])
{
        Point p1(2,4);
        p1.deplace(2,2);
        p1.affiche();
        return 0;
}
```

Exercices :

- ✓ Créez une classe **`CCompte`** permettant de gérer un compte bancaire possédant les caractéristiques suivantes :
  - Attributs :
    - **`dSolde`** ➔ attribut contenant le solde courant du compte
  - Méthodes :
    - Constructeur : Passez en paramètre le versement à la création du compte (nul par défaut)
    - **`ajouter(double dVal)`** ➔ Ajout d'une somme donnée au compte (cette somme doit être positive)
    - **`retirer(double dVal)`** ➔ Retrait d'une somme donnée au compte. L'argument doit être positif et le solde aussi
    - **`donnerSolde()`** ➔ Renvoie le solde courant

- ✓ Créez un programme principal de test gérant plusieurs comptes.