Responsable pédagogique	AF	**AM**	OP	PM
Période	Sem1	Sem2	**Sem3**	Sem4
Volume horaire	Cours/TD		TP	
			6	

[CPP2]
DU C VERS LA POO... (suite)

Indicateur temporel (hors rédaction du compte-rendu) :

questions	1h	2h	3h	4h	5h	6h
1						
2						
3..4						
5..7						
8..9						
10						

Objectifs : Passage en douceur du C vers les premiers concepts de Programmation Orientée Objet.

Documents à rendre : Compte-rendu contenant à minima les sources (avec entête standard) et les résultats obtenus.

Cahier des Charges : En vue de la gestion d'un fichier de membres d'un club sportif, on souhaite dans un premier temps manipuler des objets représentatifs des adhérents. Un tel objet, pour les besoins de l'exercice, ne contiendra ici que le nom, le prénom et la date de naissance de la personne.
Pour ce faire, la première partie (cf. sujet CtoPOO-part1) concernait le développement de 2 classes nommées CDate et CMembre, avec une relation de composition entre les 2 : l'attribut de CMembre (dateNaiss) de classe CDate.

Cette deuxième partie va permettre d'enrichir le modèle en ajoutant une classe CClubSportif et une classe CMatch.

Reprise de la première partie

1. Récupérer les fichiers `tmonth.h`, `tmonth.cpp`, `cdate.h`, `cdate.cpp`, `cmembre.h` et `cmembre.cpp` précédemment développés.
2. Vérifier le bon fonctionnement de ces ressources au moyen du programme de test suivant (corriger et/ou compléter les classes CDate et CMembre si nécessaire) :

```
#include "cmembre.h"
#include <iostream>
using namespace std ;

int main(int argc, char** argv )
{
      CMembre membre1("Tournelle", "Marie",  CDate( 1, AVR, 2000 ) ) ;

      cout << membre1.print() << endl ;
      return 0 ;
}
```

→ Résultat produit : `Marie Tournelle, né(e) le 1er avril 2000`

Création d'une nouvelle classe CClubSportif

Le rôle de cette classe va être de maintenir une liste de membres du club sportif ; le club en question est ici un club d'échecs.
Pour des raisons de structure des locaux et de créneaux horaires, le club ne peut accepter plus de 100 membres. La liste des membres sera donc matérialisée sous forme d'un attribut de type tableau de 100 pointeurs d'objets de classe CMembre.

Déclaration du tableau : `CMembre*  membres[NMAX] ;`
Initialisation : `for (int i = 0 ; i < NMAX ; ++i )  membres[i] = 0 ;`

Pour le moment, le seul autre attribut de la classe CClubSportif sera un `string` permettant de spécifier le nom du club…

3. Développer le(s) constructeur(s) de la classe en conséquence.

L'ajout d'un membre doit se faire au moyen d'une méthode publique de prototype :
`bool ajoute(const CMembre& membre ) ;`

4. Implémenter cette méthode en respectant le canevas suivant :
   - recherche du premier emplacement libre dans le tableau `membres` (→ indice `iLibre`) ;
   - création et stockage du nouvel élément : `membres[iLibre] = new CMembre(membre) ;`
   - retour d'une valeur fausse si le tableau est plein et que le nouveau membre n'a pu être créé.

5. Proposer une nouvelle méthode publique `liste()` permettant d'afficher sur la sortie standard la liste des membres enregistrés (prénom, nom, date de naissance) dans l'ordre de leurs enregistrements.

6. Développer un petit programme de test permettant d'alimenter un objet CClubSportif avec une dizaine de membres, puis d'en afficher la liste.

7. Mettre en place un destructeur assurant le nettoyage des objets recensés par le tableau des membres.

Gestion des rencontres sportives

Pour faire simple, une rencontre sportive sera caractérisée par une date et deux adversaires membres du club.

8. Imaginer le contenu d'une classe CMatch permettant de modéliser une rencontre sportive.

9. Proposer un programme de test permettant d'illustrer de manière simple la création d'une rencontre puis l'affichage de ses attributs.

10. Pour finir, dessiner un diagramme de classes montrant les classes CDate, CMembre, CClubSportif et CMatch, leurs attributs et méthodes, ainsi que leurs relations.