Examen : Programmation orientée objets en C++

Exercice 1 : (5 points)

1. Qu'est-ce que Pointeur ?
2. Qu'est-ce que l'héritage?
3. Comment déclare qu'une classe B dérive d'une classe A ?
4. Complétez le tableau suivant par : attributs et méthodes.

Droit d'accès	**Public**	**Protected**	**Private**
Classe mère			
Classe indépendante			

5. Si on implémente un constructeur au niveau de la classe fille, est ce que le constructeur de la classe mère est systématiquement appelé en premier ?

Exercice 2: (15 points)

1. On voudrait gérer les étudiants de notre département génie électrique à l'aide d'une classe **Etudiant** définie par :
Les attributs suivants :

- **nom** : nom d'un étudiant
- **prenom**: prénom d'un étudiant
- **tabnotes** : tableau contenant les notes d'un étudiant, sachant qu'un étudiant a au total 10 notes.

Les méthodes suivantes :

- **void saisie ()**, permettant la saisie d'un étudiant
- **void affichage ()**, permettant l'affichage d'un étudiant
- **float moyenne ()**, retourne comme résultat la moyenne des notes d'un étudiant.
- **int admis ()**, retourne comme résultat la valeur 1, si un étudiant est admis et la valeur 0, sinon. Un étudiant est considéré comme étant admis lorsque la moyenne de ses notes est supérieure ou égale à 10.

Écrire la classe Etudiant (Etudiant.h et Etudiant.cpp) dans le langage C++.

2/ On voudrait maintenant représenter, à l'aide d'une nouvelle classe Etudiant_Master_2 (Etudiant en Master 2). Ces étudiants possèdent en effet un attribut supplémentaire : note_memoire, qui représente la note de leur mémoire de fin d'études.
Les méthodes à associer à cette classe sont les suivantes :

- **void saisie_note_memoire ()**, permettant la saisie
- **void affichage (),** permettant l'affichage d'un étudiant en Master 2

Ecrire la classe Etudiant_Master_2 (Etudiant_Master_2.h et Etudiant_Master_2.cpp) dans le langage C++.

Exercice 1 : (5 points)

1. Un pointeur est une variable qui stocke l'adresse d'une autre variable. **(1)**

2. L'héritage c'est une technique qui permet de créer une classe à partir d'une autre classe. Cela permet d'éviter d'avoir à réécrire un même code source plusieurs fois **(1)**

3. La classe B dérive d'une classe A **(1)**

```
#include "A.h"
class B : public A
{
};
```

4. Complétez le tableau suivant par : attributs et méthodes. **(1)**

Droit d'accès	**Public**	**Protected**	**Private**
Classe mère	méthodes	attributs	
Classe indépendante	méthodes		attributs

5. OUI **(1)**

Exercice 2 : (15 points)

Etudiant.h

```
#ifndef DEF_ETUDIANT
#define DEF_ETUDIANT
#include <iostream>                    (0.5)
#include <string>
class Etudiant (0.5)
{
   public:    (0.25)
         Etudiant ();   (0.25)
         void saisie () ; (0.25)
         void affichage () ; (0.25)
         float moyenne() ; (0.25)
         int admis() ;
  protected: (0.25)
         std::string m_nom ; (0.25)
         std::string m_prenom[50]; (0.25)
         float m_tabnotes[10] ; (0.25)
};
#endif  (0.25)
```

Etudiant.ccp

```
#include " Etudiant.h"   (0.5)
using namespace std;
Etudiant:: Etudiant (std::string nom, std::string prenom, float tabnotes):m_nom (nom) ,
m_prenom (prenom), m_tabnotes(tabnotes) (1)
```

```
{
}

void Etudiant ::saisie ()  (1)

{ int i ;
        cout << "Donner le nom :" ;
        cin >> m_nom ;
        cout << "Donner le prénom :" ;
        cin >> m_prenom ;
        cout << "Saisie des notes \n" ;
        for (i = 0 ; i < 10 ; i++)
        {
                cout << "Donner la note N°" << i<< " : " ;
                cin >> m_tabnotes[i] ;
        }
}

void Etudiant ::affichage () (1)
{ int i ;
        cout << "Le nom :"<<m_nom<< endl ;
        cout << "Le prénom :" <<m_prenom<< endl ;
        for (i = 0 ; i < 10 ; i++)
                cout << "La note N°" << i << "est " << m_tabnotes[i]<< endl ;
}

float Etudiant ::moyenne()  (1)
{ int i ;
  float som = 0;
        for (i = 0 ; i < 10 ; i++)
                som + = m_tabnotes[i] ;
        return (som/10)
}

int Etudiant ::admis()   (1)

{ if (moyenne() >= 10) return (1); else return (0);}
```

Etudiant_Master_2.h

```
#ifndef DEF_ ETUDIANT_MASTER_2
#define DEF_ ETUDIANT_MASTER_2
#include <iostream>
#include " Etudiant.h"   (0.5)
#include <string>
class Etudiant_Master_2: public Etudiant (0.5)
{
   public:
Etudiant_Master_2 (); (0.5)
```

```
void saisie_note_memoire (); (0.5)
void affichage ();  (0.5)
   private:
   float m_ note_memoire;                    (0.5)
};
#endif
```

```
Etudiant_Master_2:: Etudiant_Master_2 (std::string nom, std::string prenom, float tabnotes ,
float note_memoire):Etudiant (std::string nom, std::string prenom, float tabnotes),
m_note_memoire (note_memoire) (1)
{
}

void Etudiant_Master_2:: saisie_note_memoire ()  (1)

{         Etudiant ::saisie () ;
cout << "Donner la note du mémoire :" ;
          cin >> m_ note_memoire;
}

void Etudiant_en_Maitrise ::affichage ()  (1)

{         Etudiant :: affichage () ;

cout << "La note du mémoire :" << note_memoire<< endl ;
}
```