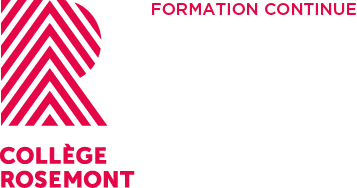
## Titre du cours

##### Concepts de la programmation orientée objet

### – Évaluation finale –

### 10 mai 2022 – Durée 4h

|  |  |
| --- | --- |
| Session Hiver 2022 | Code du cours 420-935-RO |
| Groupe 38803 | Enseignant Naïm Himrane |



### Questions théoriques : (40 points)

1. Qu’est une classe, une classe abstraire, une interface? (6 pts)

Une classe est l'unité de base de la programmation orientée objet. Est un type de données de référence créé sur la base de la similitude des choses. Contient des variables (données) et des méthodes. Les instances de classes sont des objets.

-----------------------------------------------------------------------------------------------------------------------Une classe abstraite est une autre abstraction de la communauté des caractéristiques des choses, mais une classe créée. Par exemple, les chiens, les chats et les hiboux peuvent être résumés en tant qu'animaux, et les hiboux peuvent être résumés en tant qu'oiseaux. Les classes abstraites sont représentées par le mot clé abstract et les classes abstraites ne peuvent pas être instanciées. Seuls les paramètres et méthodes concrets peuvent être déclarés. Besoin d'hériter de sa classe comme chat, hibou à atteindre. Les classes héritées doivent implémenter toutes les méthodes abstraites déclarées dans la classe abstraite.

Sauf que la classe abstraite ne peut pas instancier l'objet, d'autres fonctions de la classe existent toujours et les méthodes d'accès des variables membres, des méthodes membres et des constructeurs sont les mêmes que celles des classes ordinaires.

Les interfaces elles-mêmes sont des classes abstraites. C'est une collection de méthodes abstraites. Utilisé pour représenter la communauté des choses, un concept plus abstrait. Une interface ne déclare que des méthodes abstraites, pas de constructeurs. Les variables doivent également être des constantes statiques finales immuables.

Les interfaces ne peuvent pas non plus être instanciées. Elle doit être concrètement implémentée par la classe héritée.

Par exemple : nous pouvons créer une interface pour les oiseaux et déclarer que tous les oiseaux peuvent voler et crier.

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

1. Expliquer le mécanisme de l’héritage en orienté objet. (2 pts)

Le mécanisme d'héritage des classes varie d'un langage à l'autre. Le mécanisme d'héritage de C# est le suivant

1. Une classe ne peut hériter que d'une seule classe ; A:B

2. Une classe peut hériter de la classe parent et être héritée par la classe enfant en même temps ; A : B, B : C ...

3. Une classe peut hériter de plusieurs interfaces.

4. Lorsqu'une classe hérite de la classe parent, les propriétés et méthodes déclarées par la classe parent seront automatiquement héritées en tant que propriétés et méthodes de la classe enfant.

5. Lorsqu'une sous-classe hérite de la superclasse, elle peut remplacer la méthode de la superclasse virtuelle pour modifier la fonction de la même méthode. C'est ainsi que le polymorphisme est implémenté.

6. La sous-classe hérite des méthodes de la classe parente et peut être appelée via base.

-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

1. Qu’est ce qu’une instance ? (2 pts)

-----------------------------------------------------------------------------------------------------------------------Une instance est une implémentation concrète d'une classe, par exemple, une instance de chat est une civette noire et blanche. Un exemple de voiture est une certaine carola de Toyota2022.

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

1. C’est quoi le rôle du constructeur et du destructeur. (2 pts)

-----------------------------------------------------------------------------------------------------------------------Un constructeur est une méthode d'implémentation indispensable pour chaque classe afin de créer de nouveaux objets de cette classe. Ne peut avoir aucun paramètre et peut avoir un nombre différent de paramètres. Lorsqu'une nouvelle instance est déclarée, les membres de la classe sont automatiquement créés en fonction du constructeur concret appelé.

Les destructeurs sont tout le contraire. Lorsque l'objet (instance) est révoqué et que le code de nettoyage est ajouté, vous devez définir votre propre destructeur.

-----------------------------------------------------------------------------------------------------------------------

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

1. Lorsqu’on dérive d’une classe privée, les membres privés de la classe de base deviennent des membres privés de la classe dérivée (enfant) : (2pts)

REP: Vrai：

Premièrement, les membres privés de la classe de base sont hérités par les classes dérivées, mais ne sont pas accessibles par les classes dérivées.

Pour les variables privées membres du type privé de la classe de base, les fonctions membres définies dans la classe dérivée ne peuvent pas accéder directement aux membres privés de la classe de base, mais ne sont accessibles que via les fonctions membres public ou protect de la classe de base.





1. Lequel des énoncés suivants concernant l’héritage multiple en C# est vrai ? (une réponse est correcte) (4 pts)

LA REPONSE EST:







1. Sélectionner les énoncés qui sont corrects : (une réponse est correcte) (4 pts)









1. Sélectionner les énoncés qui sont corrects : (on peut avoir plusieurs réponses) (4 pts)

LA REPONSE EST:

1）et 4）

\*





\*

1. Les variables et méthodes statiques : (on peut avoir plusieurs réponses) (4 pts)

LA REPONSE EST : 1) ET 2) ET 4)

\*

\*



\*

1. En considérant les deux interfaces suivantes qui ont une méthode avec le même nom et la classe qui implémente les deux : (10 pts)

public interface IConduire

{

void Demarrer();

void Conduire();

}

public interface IVoler

{

void Demarrer();

void Voler();

**}**

publicclassMobilite **:** IConduire**,** IVoler

**{**

publicvoidConduire**() {** Console**.** WriteLine **(**"Conduire"**);}**

publicvoidVoler**() {** Console**.** WriteLine **(**"Voler"**);}**

**}**

Quelles méthodes doivent être ajoutées dans la classe « Mobilite » afin d’obtenir le résultat suivant :

IVoler.Demarrer

Voler

IConduire.Demarrer

Conduire

LA REPONSE EST : 1)

\*

void IConduire.Demarrer()

{

Console.WriteLine("IConduire.Demarrer");

}

void IVoler.Demarrer()

{

Console.WriteLine("IVoler.Demarrer");

}



public void IConduire.Demarrer()

{

Console.WriteLine("IConduire.Demarrer");

}

public void IVoler.Demarrer()

{

Console.WriteLine("IVoler.Demarrer");

}



public void Demarrer()

{

If (this is IVoler)

{

Console.WriteLine("IVoler.Demarrer");

}

else if (this is IConduire)

{

Console.WriteLine("IConduire.Demarrer");

}

}



### Questions pratiques : (60 points)

1. Créer un projet en .NET Core avec le nom « ***Evaluation3\_Votre\_Prenom*** » (Mon prénom est Naim : ***Evaluation3\_Naim***). Votre projet est de type Console C#.
2. Créer et implémenter toutes les classes et interfaces comme le montre le schéma suivant :



1. Respecter les notions de l’héritage et du polymorphisme;
2. Ajouter les méthodes qui manipulent vos données membres;
3. Utiliser une variable « static  nombre\_vehicules » qui nous retourne le nombre de véhicules construits jusqu’à date;
4. À chaque arrêt d’un véhicule, vous devez mémoriser et afficher le nombre de kilométrage; vous aurez besoin d’ajouter la variable membre qui calcule le nombre de kilomètres;
5. Créer une liste de véhicules qui nous permet d’afficher la marque et l’accélération de chaque type de véhicule créé; vous aurez besoin d’ajouter la variable membre pour l’accélération;