.NET Technology

(503112)

By Mai Van Manh

Lab 02: Object Orieted in C#

Mỗi bài học thực hành đều đi kèm với một số tập tin PDF hướng dẫn các kiến thức có liên quan đến các bài tập. Sinh viên cần xem các tập tin hướng dẫn (và video nếu có) trước khi thực hiện các bài tập bên dưới. Sinh viên chỉ nộp lại source code (.cs, .txt; và static contents, media) của từng bài tập, không nộp toàn bộ project. Với bài tập giao diện (Forms, Web) sinh viên cần cung cấp thêm ảnh screenshot từng giao diện của bài tập. Dữ liệu mỗi bài tập cần được đặt trong một thư mục riêng (trừ khi bài tập chỉ có 1 file duy nhất), toàn bộ bài tập cần được đặt trong thư mục có dạng 520H1234_NguyenVanA và được nén lại dưới dạng zip/rar trước khi nộp. Bài nộp không có thông tin sinh viên hoặc nộp toàn bộ source code của project sẽ không được tính.

Exercise 1: Library Catalog

You are tasked with creating a library catalog system in C#. Follow these requirements:

Create a **Book** class with the following attributes:

- Title
- Author
- ISBN (International Standard Book Number)
- Year of Publication

Create a **Library** class that can hold multiple books. It should have the following methods:

- **AddBook**(Book book): Adds a book to the library catalog.
- **RemoveBook**(Book book): Removes a book from the library catalog.
- **FindBookByTitle**(string title): Returns the first book found with the given title.
- **ListAllBooks**(): Displays details of all books in the library catalog.

Create a **Person** class to represent library patrons. Each person should have a name and an age.

Create a **LibraryApp** class with a **Main** method to test your library system. Create instances of books, add them to the library, and demonstrate the library's functionality by listing books, adding books, and removing books.

Exercise 2: Online Store

Create a **Product** class with the following attributes: ProductID, Name, Price, Description.

Create a **Customer** class with the following attributes: CustomerID, Name, Email, Address.

Create an **Order** class to represent customer orders. Each order should have: OrderID, Customer, List of products, Order date.

Implement a **ShoppingCart** class to allow customers to add and remove products before placing an order. Create a **StoreApp** class with a Main method to test the online store system. Create instances of products and customers, add products to a shopping cart, simulate placing an order, and display order details.

Exercise 3: Advanced Zoo Management System

Imagine you're responsible for developing an advanced Zoo Management System that handles a wide variety of animals and provides versatile functionality for managing them. You'll need to implement classes, abstract classes, interfaces, and various methods to achieve this goal.

Animal:

- Create an abstract class named **Animal**.
- Include properties for **Name**, **Age**, and **Species**.
- Include a method **MakeSound**() that prints a generic sound of the animal.

Mammal:

- Create a class named **Mammal** that inherits from **Animal**.
- Add a property **NumberOfLegs** specific to mammals.
- Implement the **MakeSound**() method to print a mammal-specific sound.
- Introduce an abstract method **GiveBirth**() to indicate that mammals give birth.

Bird:

- Create an interface named **IFlyable** with a method **Fly**() without implementation.
- Create a class named **Bird** that inherits from **Animal** and implements the **IFlyable** interface.
- Include a property **Wingspan** specific to birds.
- Implement the **MakeSound**() method to print a bird-specific sound.
- Implement the **Fly**() method to print a flying message.

Reptile:

- Create a class named **Reptile** that inherits from Animal.
- Add a property **IsVenomous** specific to reptiles.
- Implement the **MakeSound**() method to print a reptile-specific sound.

ZooKeeper:

- Create a class named **ZooKeeper**.
- Include a private **list of Animal** objects to manage animals in the zoo.
- Implement methods:

- **AddAnimal**(Animal animal): Adds an animal to the zoo.
- **RemoveAnimal**(string animalName): Removes an animal from the zoo based on its name.
- **FindAnimal**(string animalName): Returns the Animal object with the given name.
- **FilterAnimals**(Func<Animal, bool> criteria): Returns a list of animals that satisfy the given criteria function.

Main Method:

- Create a Main method to demonstrate your advanced Zoo Management System.
- Instantiate various animals of different types (mammals, birds, reptiles) and showcase their attributes and behaviors.
- Utilize the ZooKeeper class to manage the animals, including adding, removing, finding, and filtering based on criteria.