# Lab: Inheritance

Problems for exercises and homework for the ["C# OOP" course @ SoftUni".](https://softuni.bg/trainings/4697/csharp-oop-october-2024)

You can check your solutions here: [Inheritance-Lab](https://alpha.judge.softuni.org/contests/1499)

# Part I: Inheritance

## Single Inheritance

**NOTE**: You need a public StartUp class with the namespace **Farm**.

Create two classes named Animal and Dog:

* Animal with a single public method Eat() that prints: **"eating…"**
* Dog with a single public method Bark() that prints: **"barking…"**
* Dog should inherit from Animal

|  |
| --- |
| **Sample Main()** |
| static void Main()  {  Dog dog = new Dog();  dog.Bark();  dog.Bark();  } |

### Hints

Use the **":" operator** to build a hierarchy

## Multiple Inheritance

**NOTE**: You need a public StartUp class with the namespace **Farm**.

Create three classes named Animal, Dog, and Puppy:

* Animal with a single public method Eat() that prints: **"eating…"**
* Dog with a single public method Bark() that prints: **"barking…"**
* Puppy with a single public method Weep() that prints: **"****weeping…"**
* Dog should inherit from Animal
* Puppy should inherit from Dog

|  |
| --- |
| **Sample Main()** |
| static void Main()  {  Puppy puppy = new Puppy();  puppy.Eat();  puppy.Bark();  puppy.Weep();  } |

## Hierarchical Inheritance

**NOTE**: You need a public StartUp class with the namespace **Farm**.

Create three classes named Animal, Dog, and Cat:

* Animal with a single public method Eat() that prints: **"eating…"**
* Dog with a single public method Bark() that prints: **"barking…"**
* Cat with a single public method Meow() that prints: **"meowing…"**
* Dog and Cat should inherit from Animal

|  |
| --- |
| **Sample Main()** |
| static void Main()  {  Dog dog = new Dog();  dog.Eat();  dog.Bark();  Cat cat = new Cat();  cat.Eat();  cat.Meow();  } |

# Part II: Reusing Classes

## Random List

**NOTE**: You need a public StartUp class with the namespace **CustomRandomList**.

Create a RandomList class that has all the functionality of List<string>. Add an additional function that **returns** and **removes** a random element from the list.

* Public method: RandomString(): string

## Stack of Strings

**NOTE**: You need a public StartUp class with the namespace **CustomStack**.

Create a class StackOfStrings that extends **Stack**, can store only strings, and has the following functionality:

* Public method: IsEmpty(): bool
* Public method: **AddRange(): Stack<string>**