# Lab: Inheritance

Problems for exercises and homework for the ["CSharp DB Advanced" course @ Software University](https://softuni.bg/trainings/1741/databases-advanced-entity-framework-october-2017).

# Part I: Inheritance

## Single Inheritance

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…"**

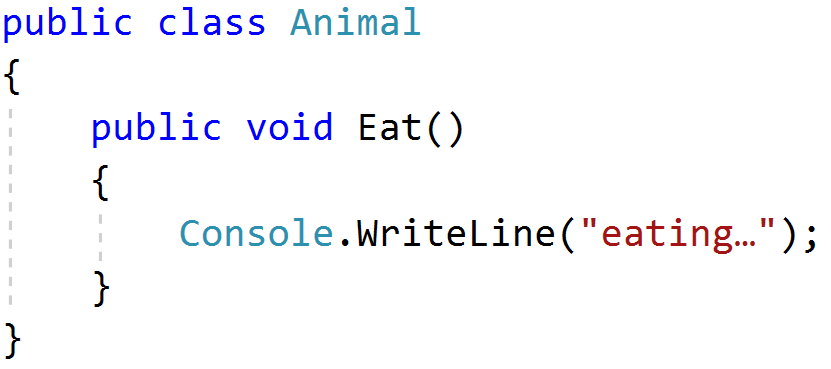
Dog should inherit from Animal.

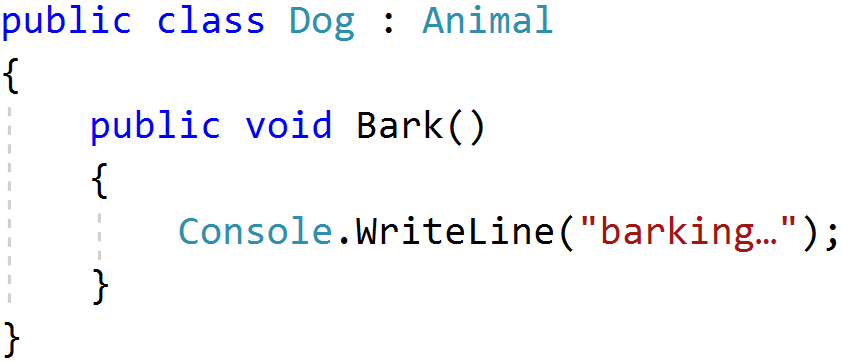


### Hints

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

### Solution





## Multiple Inheritance

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…"**

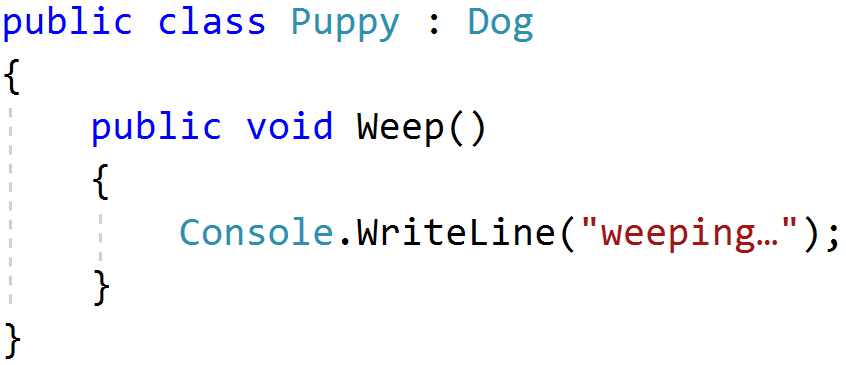
Puppy with a single public method Weep() that prints: **"****weeping…"**

Dog should inherit from Animal. Puppy should inherit from Dog.



### Solution

Add a new class Puppy to your project from task 1:



## Hierarchical Inheritance

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…"**

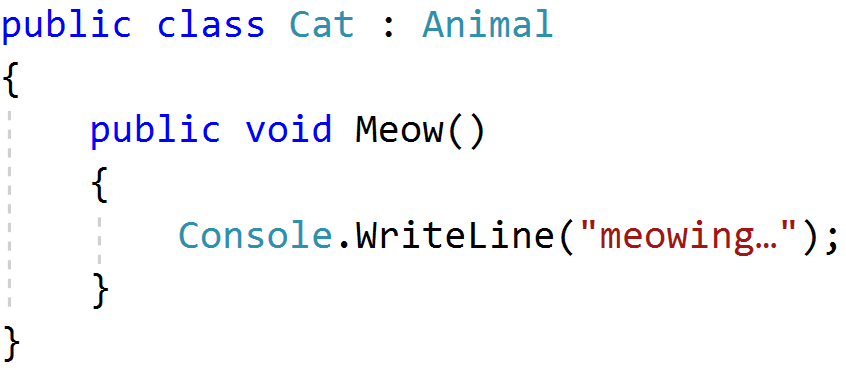
Cat with a single public method Meow() that prints: **"meowing…"**

Dog and Cat should inherit from Animal.



### Solution

Add a new class Cat to your existing project that inherits the Animal class:



## Random List

Create a RandomList class that has all the functionality of List<string>.

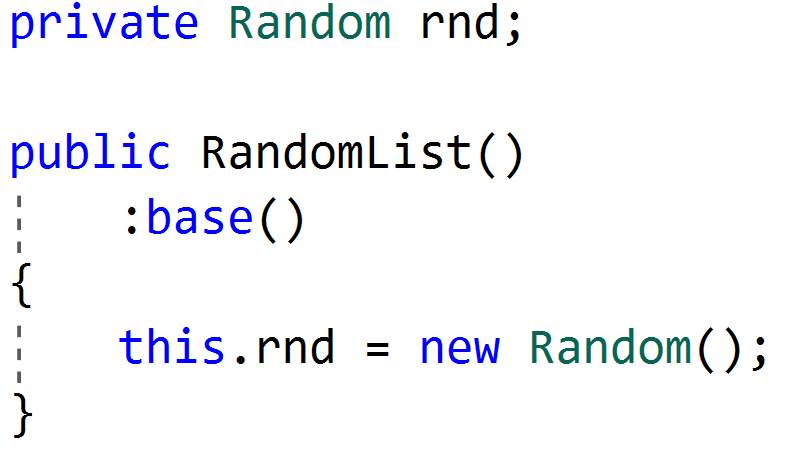
Add additional function that **returns** and **removes** a random element from the list.

* Public method: RandomString(): string

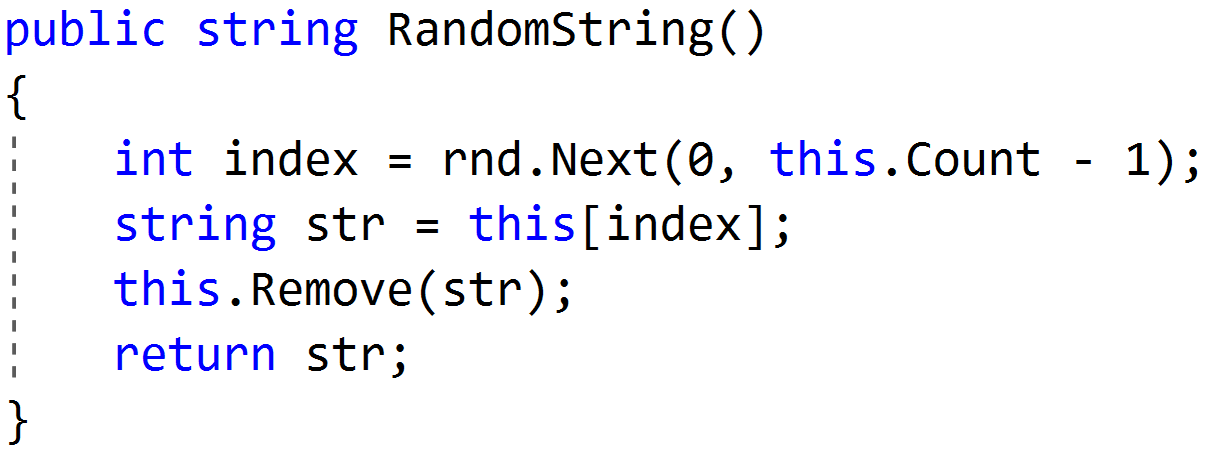
Create a class RandomList, derived from List<string>:

****

Create a Random object as a property, a constructor that calls the base one and initialises the Random object:



And then create the RandomString() method:



## Storage Box

Create a generic class Box that can be initialized with **any** type and **store** values. **Override** the **ToString()** method to print the values stored, separated by a **comma** and a **space** (", ").

The Box class should have:

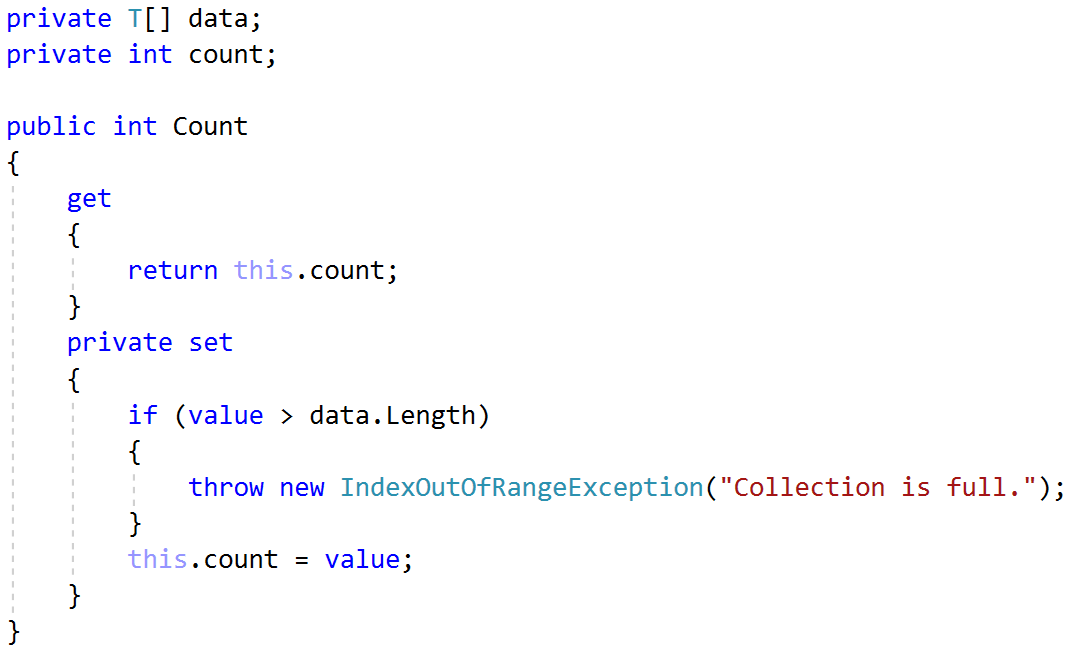
* Private field: data: generic array that stores the inserted elements
* Private field: count: intger, holding the count of elements in the Box
* A constructor and the necessary getters and setters
* Public method: Add(): void – adds an element to the Box after checking if there is space left in the array. If it’s full, it must copy the data to a new array which is twice the size of the previous one.
* Public method: Remove(): generic – removes and returns the last added element from the Box
* Public override of ToString()

You should be able to use the class like this:

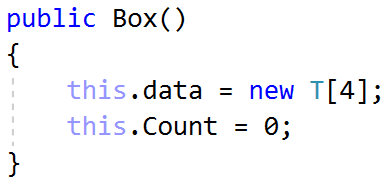
|  |
| --- |
| public static void Main()  {  var box = new Box<int>();  for (int i = 0; i < 11; i++)  {  box.Add(i);  }  System.Console.WriteLine(box);  } |

### **Solution**

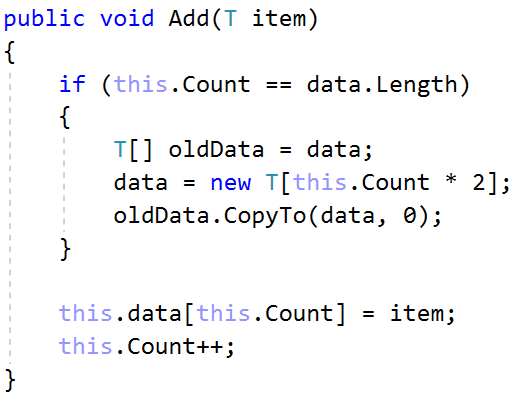
Create the **fields** and **properties** we will need. Validate count in the setter, so it can never be more than the Length of the data array.

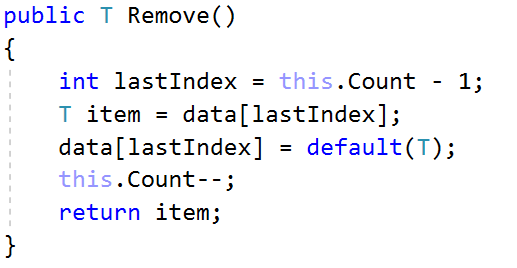


Create a **constructor** that gives the Box an empty data array with a size of 4 and a Count of 0.



Next, create the Add() and Remove() methods:





Last, override the ToString() method:

