声明：有些字符串因排版要求换行，正常使用应在前加上“@”，或保持在同一行

继承与实现

基础篇

要求：
1、明确什么是继承
2、熟练的实现继承，写出类的代码，并实例化

为什么要有继承

前面我们已经学会了如何写一个类，并且对其进行实例化和使用. 下面我们来看看如果有这样一个情景怎么处理：

要求写一个学生类、老师类、家长类和一个校长类. 其中四个类中都有“姓名”、“性别”和“年龄”，但是四个类都还具有特定属性. 例如学生有爱好，老师有工作年限、教学科目，家长有工作，校长有职称等. 同时，每一个类的对象都有自我介绍的方法，只是每个自我介绍的方法不太一样.

那么如何完成呢？

很显然此时按照前面类的写法，四个类中都要写姓名、性别和年龄的字段和属性. 对此代码的重复量实在是太大，那么可以想：

如果能写一个具有姓名、性别、和年龄的字段和属性的类，创建这四个类的时候，将其包含进来就好了.

Bingo！这就用到了继承，使用继承的方法：

写一个类，包含姓名、性别、年龄的字段和属性，然后再写学生、老师、家长与校长类，将其包含进去，那么这些类就都具有了姓名、性别和年龄了.

如何实现继承

下面为了简单说明继承的语法和实现，我们用最简单类来描述，再回到一开始说的例子当中来.

1、写一个父类

父类就是那个被包含的类，在里面可以写上其他类都有的字段. 可以这么写：

```
class MyBase
{
    public int numBase = 10;
}
```

注：

-> 只需要在一个类里面写上成员即可，目的是让别的类包含它
-> 如果在后面的类中没有写 numBase，但能够访问这个字段，就表示继承了
-> 这个类叫做父类，或称为基类
-> 这里为了说明问题，使用 public 进行修饰，目的是为了使类的结构最简单

2、继承的语法

继承是一个过程. 就是再写一个类，使这个类包含刚才 Base 类中的成员. 包含使用“:”语法，代码可以写成这样：

```
class MySub : MyBase
{
    public int numSub = 20;
}
```

注：

-> 这个类的写法与其他类似乎一样，但是在类名后面多了“: MyBase”

-> 多了这个东西，就表示 MySub 这个类就包含了 MyBase 这个类

-> 此时这个类叫做子类，或派生类

3、实例化子类

实例化很简单，就是 new 了. 在 Main 方法中，实例化 MySub 对象，添加如下代码：

```
static void Main(string[] args)
{
    MySub mySub = new MySub();
}
```

那么就实例化了一个子类对象，然后使用 mySub 对象，看看可以点出什么？

```
static void Main(string[] args)
{
    MySub mySub = new MySub();
    Console.WriteLine(mySub.numBase);
    Console.WriteLine(mySub.numSub);
    Console.ReadKey();
}
```

注：

-> 此时运行结果是 10 和 20

-> 子类 MySub 中没有添加 numBase 这个字段，但是依旧可以访问到，说明继承了

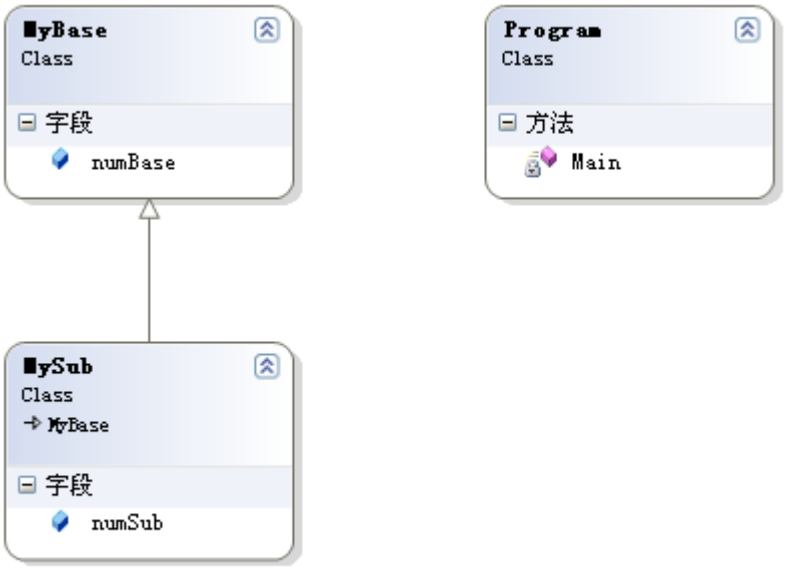
-> 应该注意一下，这里为了说明问题使用的 public 修饰，实际中应该考虑使用属性

上面的代码就是对继承的一个很好的描述，此处 MyBase 称为“父类”，好比孩子继承父亲的遗传一样，具有父亲的特征，这里将这个类称为父类，就是说明其他继承它的类都具有其包含的特征（这里应该注意访问修饰符）. 而 MySub 称为子类，表示从 MyBase 继承而来的类型. 子类中除了包含父类的成员外，还可以具有自己特有的成员，当然这个成员父类是不具备的. 就好比 MySub 中 numSub 一样，是子类特有的. 可以实例化一个 MyBase 类型出来看看，是没有 numSub 可以访问的.

这里父类也被称为“基类”，子类也成为“派生类”. MySub 继承于 MyBase，也称为 MyBase 派生出 MySub.

继承的类图

通过 VS 的类图查看功能可以得到上述代码的类图




注：

-> 类图查看是选中项目，在资源管理器上会出现快捷方式

-> 或者直接右键项目，查看类图

-> 类图使用一个单向箭头线段描述继承关系，箭头从子类指向父类（容易被忽略）

比较正规的写法

对于刚才的例子仍然是有问题的，因为咱知道字段要设定访问级别，而刚刚设为 public 是为了方便描述，下面考虑一下将刚才的例子写得完整一点：

```
// 定义父类
class MyBase
{
    private int numBase = 10;
    public int NumBase
    {
        get {
            return numBase;
        }
        set {
            numBase = value;
        }
    }
}
// 定义子类
class MySub : MyBase
{
    private int numSub = 20;
    public int NumSub
```

```
        {
            get {
                return numSub;
            }
            set {
                numSub = value;
            }
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            MySub mySub = new MySub();
            Console.WriteLine(mySub.NumBase);
            Console.WriteLine(muSub.NumSub);
            Console.ReadKey();
        }
    }
```

注：

-> 此时定义字段为私有，提供属性让外界对其进行访问

-> Main 方法中使用属性访问两个字段，输出两个字段的值

练习时间

考虑开始提到的问题：写一个 Student 类和一个 Teacher 类,他们都有一个打招呼的方法,不同的是 Studetn 打招呼是说"大家好,我叫 XX,我今年 XX 岁了,我的爱好是 XXX",Teacher 的打招呼的方法是说"大家好,我叫 XX,我今年 XX 岁了,我已经工作 XX 年了"?

1、新建一个项目，命名为“Person 练习”

2、添加父类.

在项目资源管理器中，选择项目 -> 添加 -> 类 -> 命名为 Person，修改代码：

```
    class Person
    {
        private string _name;
        private int _age;
        private char _gender;

        public string Name {
            get {
                return _name;
            }
            set {
                _name = value;
```

```
            }
        }
        public int Age {
            get {
                return _age;
            }
            set {
                _age = value;
            }
        }
        public char Gender {
            get {
                return _gender;
            }
            set {
                _gender = value;
            }
        }
    }
```

3、添加 Student 类

同步骤 2，添加一个名为 Student 的类文件，修改代码：

```
    class Student : Person
    {
        private string _hobby;
        public void SayHello() {
            Console.WriteLine("大家好，我是{0}，我今年{1}岁了，我的爱好是{2}",
                    Name, Age, _hobby
                );
        }
        public string Hobby {
            get {
                return _hobby;
            }
            set {
                _hobby = value;
            }
        }
    }
```

4、添加 Teacher 类

同理，修改代码：

```
    class Teacher : Person
    {
        private int _yearsOfService;
        public void SayHello() {
```

```
            Console.WriteLine("大家好，我叫{0}，我今年{1}岁了，我已经工作{2}年了",
                    Name, Age, _yearsOfService
                )
        }
        public int YearsOfService {
            get {
                return _yearsOfService;
            }
            set {
                _yearsOfService = value;
            }
        }
    }
```

5、在 Program.cs 文件中的 Main 方法里添加代码

```
    static void Main(string[] args)
    {
        Student stu = new Student();
        stu.Name = "张三";
        stu.Age = 18;
        stu.Gender = '男';
        stu.Hobby = "打篮球";
        stu.SayHello();

        Teacher th = new Teacher();
        th.Name = "王五";
        th.Age = 45;
        th.Gender = '男';
        th.YearsOfService = 20;
        th.SayHello();
    }
```

6、编译、运行结果为：

大家好，我是张三，我今年 18 岁了，我的爱好是打篮球

大家好，我叫王五，我今年 45 岁了，我已经工作 20 年了

注：

-> 实例化两个类，并调用各自的方法，而两个类具有相同的字段，因此写父类

-> Main 方法中，先实例化 Student，并为其姓名、性别、年龄、和爱好初始化. 再调用方法

-> 再实例化 Teacher，同样对其进行初始化，后调用方法.