继承

- 继承（以及封装和多态性）是面向对象的编程的三个主要特征之一
- 通过继承，可以创建新类，以便重用、扩展和修改在其他类中定义的行为
- 其成员被继承的类称为“基类”，继承这些成员的类称为“派生类”。 派生类只能有一个直接基类
- 但是，继承是可传递的，如果 ClassC 派生自 ClassB，并且 ClassB 派生自 ClassA，则 ClassC 将继承在 ClassB 和 ClassA 中声明的成员

```
//订单类
class Order {
    protected string name;
    protected int num;
    protected float price;

    public Order(string name, int num, float price) {
        this.name = name;
        this.num = num;
        this.price = price;
    }

    public float TotalPrice {
        get {
            return price * num;
        }
    }
}

//打折订单类，用派生类类名":"基类类名
class DiscountOrder : Order {
    public float discount = 0.8F;

    //子类特有的初始化:base(),构建派生类对象要先去构建属于基类的部分
    public DiscountOrder(string name, int num, float price, float discount) :
base(name, num, price) {
        this.discount = discount;
    }

    public float TotalPrice {
        get {
            return price * num * discount;
        }
    }
}

//调用类
 Order order1 = new Order("苹果", 10, 1);
 DiscountOrder order2 = new DiscountOrder("梨子", 5, 3, 0.5F);
 Console.WriteLine(order1.TotalPrice);
 Console.WriteLine(order2.TotalPrice);
```

```
结果：
10
7.5
```

多态

- 基类变量可以指向派生类的对象
- 通过多态，在通过基类变量调用方法时，也能自动匹配到派生类的方法上去，这就叫做多态

```
//基类对象可以指向派生类的对象，但是要重写
//不重写
Order temp = order2;
Console.WriteLine(temp.TotslPrice);

结果：
15//它调用的是基类的方法

//使用虚方法virtual和override重写基类
class Order {
    protected string name;
    protected int num;
    protected float price;

    public Order(string name, int num, float price) {
        this.name = name;
        this.num = num;
        this.price = price;
    }
    //在需要重写的方法中添加virtual
    public virtual float TotalPrice {
        get {
            return price * num;
        }
    }
}

class DiscountOrder : Order {
    public float discount = 0.8F;

    //子类特有的初始化:base(),构建派生类对象要先去构建属于基类的部分
    public DiscountOrder(string name, int num, float price, float discount) :
base(name, num, price) {
        this.discount = discount;
    }
    //派生类中添加override
    public override float TotalPrice {
        get {
            return price * num * discount;
        }
```

```
    }
}

Order temp = order2;
Console.WriteLine(temp.TotslPrice);

结果：
7.5
```