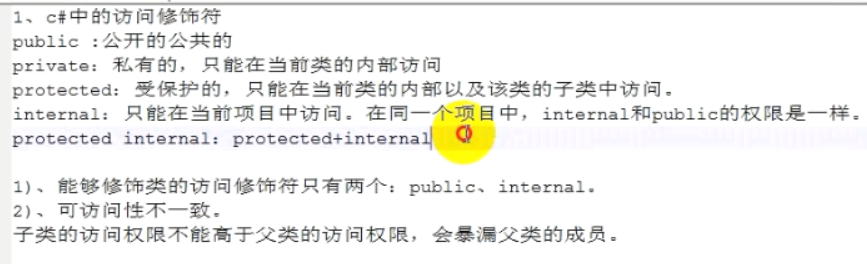
# 2、c#中的访问修饰符

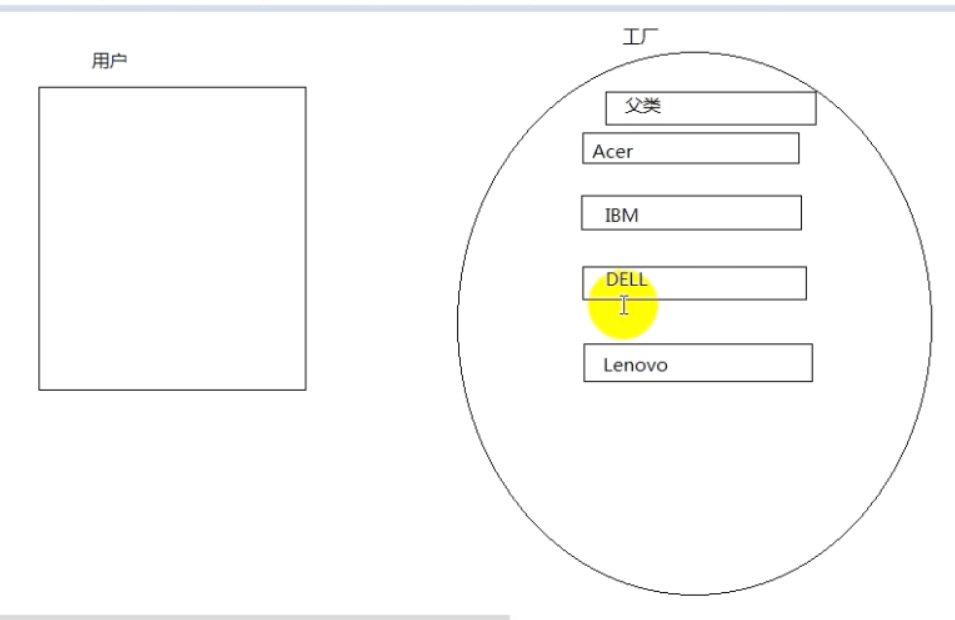
1. c#中的访问修饰符



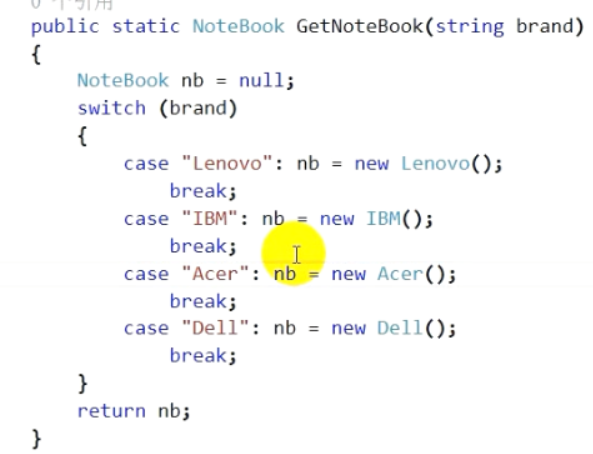
# 3、简单工厂设计模式

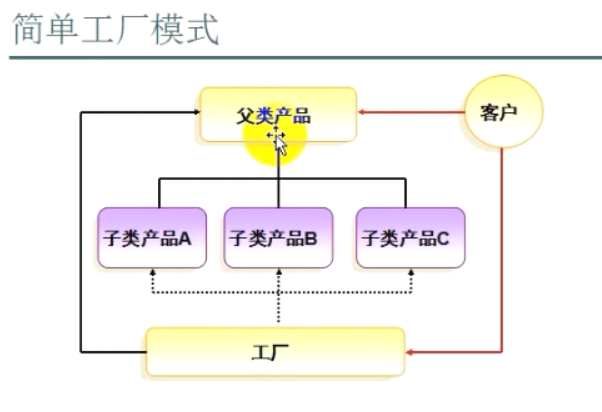
1.设计模式:设计项目的一种方式.

2.简单工厂设计模式:





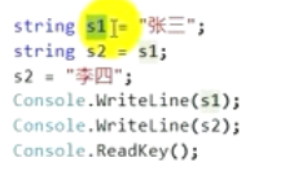


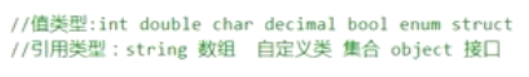


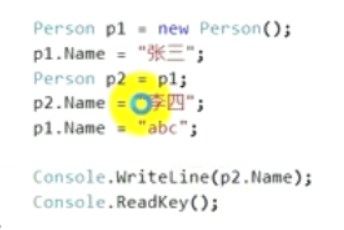
# 4、值传递和引用传递

1.值类型存储在栈上,引用类型存储在堆上.

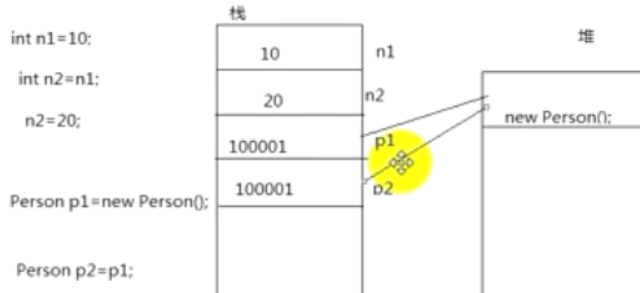
值在赋值时,赋的是值,但是引用类型赋值时,赋的是引用(也就是地址),虽然string类型也是引用类型,但是根据string类型的不可变性,string类型在赋值时,赋的不是引用而是重新开辟堆中的空间,

所以: 输出结果是:张三李四



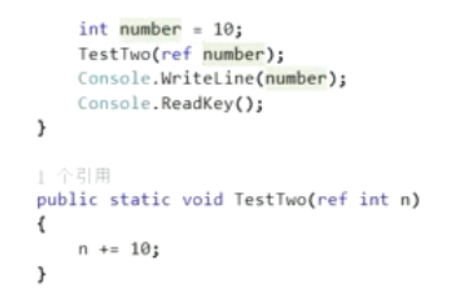


没有p1.Name=”abc”时,输出李四,如果有p1.Name=”abc”时则输出abc.

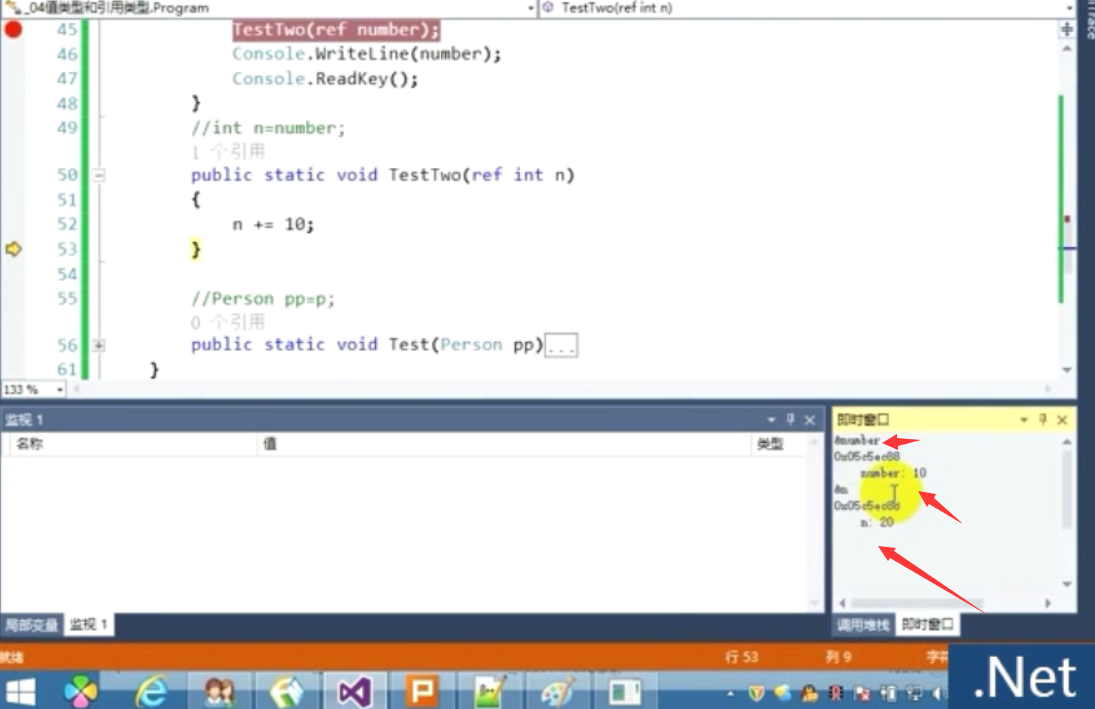


2.ref的作用:ref把一个值已参数的形式带到方法中去改变,再将改变后的值返回回来

即:ref的作用就是使两个值在栈上的地址由不一样变为一样



输出20,如果不加ref就输出10.



# 5、序列化和反序列化(没看)

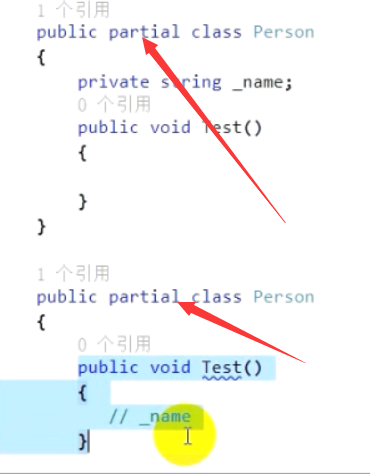
//序列化：就是将对象转换为二进制

//反序列化：就是将二进制转换为对象

//作用：传输数据 01010101 1 0

# 6、部分类

1.加关键字:

partial就可以使多个人同时对一个类进行编

# 7、密封类

1.加sealed关键字,可以使类密封.那样就不能够被其他类继承,但是可以继承于其他类.

# 8、重写父类的ToString()(Object的虚方法)

1.如果直接打印一个对象的toString,则输出这个对象类的命名空间.

# 9、接口简介

1.接口:是一种规范,只要一个类继承了一个接口,这个类就必须实现这个接口中的所有成员.

接口的语法:[public] interface I..able

{

}

2.接口的特点:

1.接口中的成员不允许添加访问修饰符,默认就是public.

2.接口中的成员不允许写具体的方法体的函数

3.接口中不能包含字段和构造函数.

4.接口中能写自动属性(本质就是属性),但不能对属性进行限定.

5. 接口不能被实例化,也就是说接口不能new对象,因为接口中的方法没有方法体,所以接口的实例化没有意义.

6.接口与接口之间可以继承,并且可以多继承!但是类不能多继承.

7.接口并不能去继承一个类,而类可以继承接口,接口只能继承于接口,而类既可以继承接口,也可以继承类.

8.实现接口的子类必须实现该接口的全部成员.

9.一个类可以同时继承一个类并实现多个接口,如果一个子类同时继承了父类A,并实现了接口IA,那么语法上A必须写在IA的前面.就是说,一个类同时继承一个父类和一个接口,那么父类必须写在接口的前面.eg: Class MyClass:A,IA

{

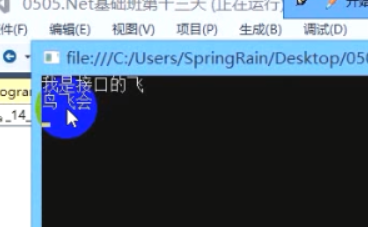
}.

10.**显示实现接口就是为解决方法的重名问题.**

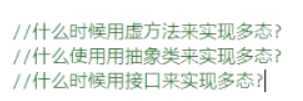




输出结果:



# 13、接口小练习



1. 什么时候用虚方法来实现多态?

答:当需要实例化父类并调用到该方法,但是这个方法在子类中又都不一样时,这时多态用虚方法来实现.例如,哈士奇会叫,金毛会叫,泰迪也会叫,这时可以用狗作为他们的父类,但是狗是个实实在在的实物,不是抽象的,它也有叫的功能(狗都会旺旺叫),所以这时就用虚方法.

1. 什么时候用抽象类来实现多态?

答:当不需要实例化父类而且该方法在父类中不能具体实现,只有在具体子类中具体实现时,这时多态就用抽象类来实现.例如: 狼会叫,狗会叫,猫也会叫,如果要实现这个叫的功能,就通过抽象个动物类的叫来实现.

1. 什么时候用接口来实现多态

答:当只是要实现不同功能,而不需要也不能提出共同的父类时就用接口来实现多态,例如:鹦鹉会飞,猫头鹰也会飞,直升机也会飞,这时候就得用接口来实现飞这个功能.