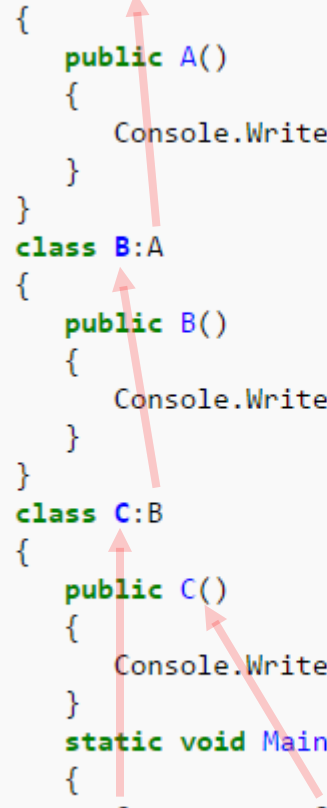


Çoklu Türetmeler

- Sınıflar tıpkı nine, anne, çocuk yapısında olduğu gibi ard arda türetilebilir. Yani örneğin B sınıfı A sınıfında türetilip C sınıfı da B sınıfından türetilebilir. Bu durumda C sınıfı türünden bir nesne yarattığımızda eğer C sınıfının ilgili yapıcı metoduna base takısını eklememişsek önce A, sonra B, sonra da C sınıfının yapıcı metotları çalıştırılır. Yani gidişat anadan yavruya doğrudur. Ayrıca tahmin edebileceğiniz gibi C sınıfı hem A'nın hem de B'nin bütün üye elemanlarına sahip olur.

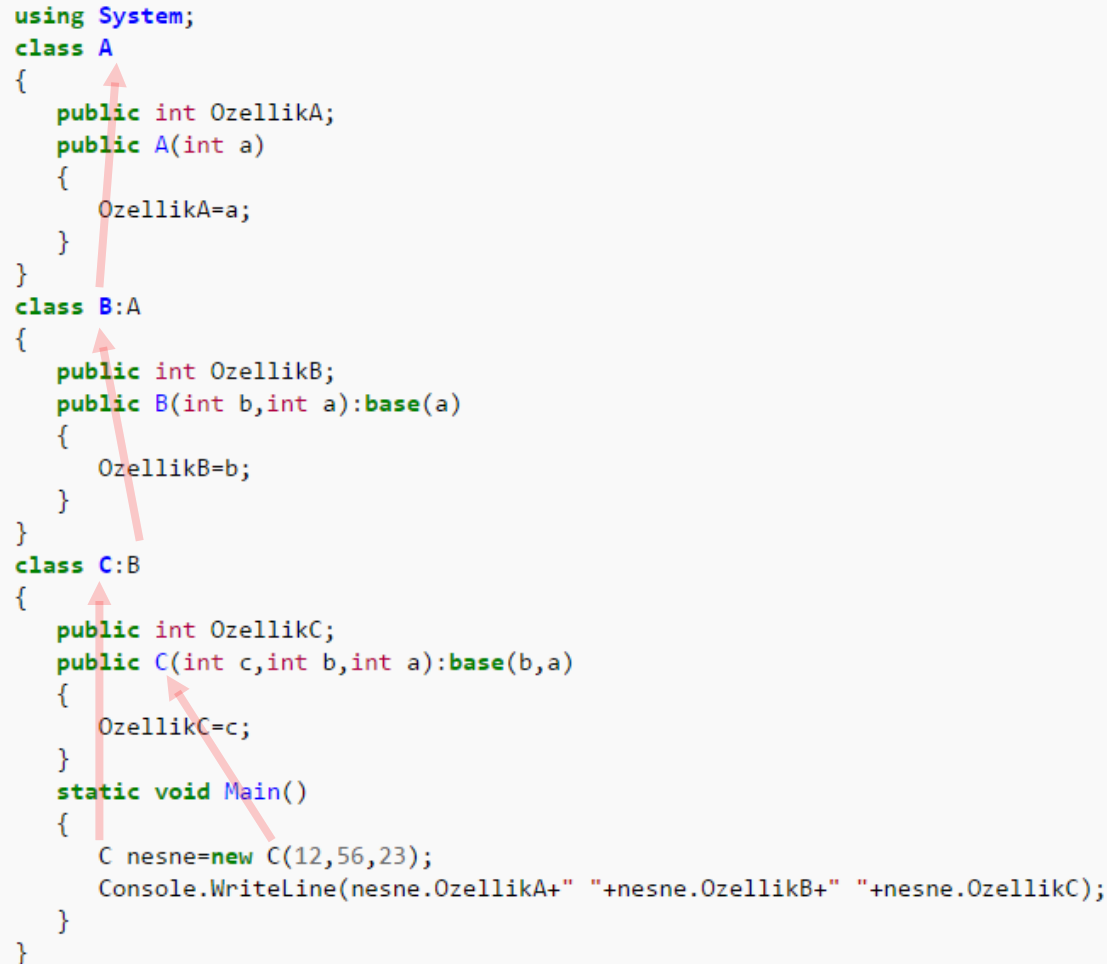
```
using System;
class A
{
    public A()
    {
        Console.WriteLine("A sınıfı");
    }
}
class B:A
{
    public B()
    {
        Console.WriteLine("B sınıfı");
    }
}
class C:B
{
    public C()
    {
        Console.WriteLine("C sınıfı");
    }
    static void Main()
    {
        C nesne=new C();
    }
}
```



Çoklu Türetmeler

- Bu program ekrana alt alta A sınıfı, B sınıfı ve C sınıfı yazacaktır. Bu örnekte base anahtar sözcüğünün kullanımı ise şöyledir:

```
using System;
class A
{
    public int OzellikA;
    public A(int a)
    {
        OzellikA=a;
    }
}
class B:A
{
    public int OzellikB;
    public B(int b,int a):base(a)
    {
        OzellikB=b;
    }
}
class C:B
{
    public int OzellikC;
    public C(int c,int b,int a):base(b,a)
    {
        OzellikC=c;
    }
    static void Main()
    {
        C nesne=new C(12,56,23);
        Console.WriteLine(nesne.OzellikA+" "+nesne.OzellikB+" "+nesne.OzellikC);
    }
}
```

The diagram illustrates the inheritance hierarchy using red arrows. One arrow points from the 'base(a)' parameter in the B class constructor to the 'A' class definition. Another arrow points from the 'base(b,a)' parameter in the C class constructor to the 'B' class definition. A third arrow points from the 'base(b,a)' parameter in the C class constructor to the 'A' class definition, indicating that C inherits from B, which in turn inherits from A.

Çoklu Türetmeler

- Gördüğümüz gibi base anahtar sözcüğü kendisinin bir üstündeki sınıfın yapıcı metoduna parametre gönderiyor ancak tabii ki işlemler alt sınıftaki üye elemanlar için yapılıyor. Bu durumda -hangi sınıf türünden nesne yaratacağımızı- hiçbir sınıfın parametre almayan yapıcı metodu çalıştırılmayacaktır (tabii ki nesne yaratırken gerekli parametreleri verdiğimiz müddetçe). Bu örneği şöyle değiştirirsek program **hata verir**:

```
using System;
class A
{
    public int OzellikA;
    public A(int a)
    {
        OzellikA=a;
    }
}
class B:A
{
    public int OzellikB;
    public B(int b)
    {
        OzellikB=b;
    }
}
class C:B
{
    public int OzellikC;
    public C(int c,int b):base(b)
    {
        OzellikC=c;
    }
    static void Main()
    {
        C nesne=new C(12,56);
        Console.WriteLine(nesne.OzellikA+" "+nesne.OzellikB+" "+nesne.OzellikC);
    }
}
```

Base şart çünkü A'da parametrelili bir constructor var.