

Türemiş sınıfta aynı metodu üretme

```
public class X
{
    public void test()
    {
        Console.WriteLine("Ben X");
    }
}

public class Y : X
{
    public void test()
    {
        Console.WriteLine("Ben Y");
    }
}

class Program
{
    static void Main(string[] args)
    {
        X _x = new X();
        _x.test();
        Y _y = new Y();
        _y.test();
        Console.ReadKey();
    }
}
```

- NYP yandaki örnek çok biçimlilik (polimorfizm).
- Ancak, klasik bir sınıftan farklı olarak türemiş sınıfta base sınıfla aynı isme, aynı parametre sayısına ve veri türüne sahip bir metod tanımlayabiliriz.
- Kodumuz çalışır. X için X'in test metodunu, Y için Y'nin X metodunu çalıştırır. Ancak altı çizili hata verir.
- Olayın farkında olduğunuzu new yazıp berlitebilirsiniz. (Değişkenlerde benzer bir durum yapmıştık. – Geçen hafta)

Türemiş sınıfta aynı metodu üretme

```
public class X
{
    public void test()
    {
        Console.WriteLine("Ben X");
    }
}

public class Y : X
{
    new public void test()
    {
        Console.WriteLine("Ben Y");
    }
}

class Program
{
    static void Main(string[] args)
    {
        X _x = new X();
        _x.test();
        Y _y = new Y();
        _y.test();
        Console.ReadKey();
    }
}
```

- Yeşil hatalar yanlışlıkla hata yapabileceğiniz anlamında karşınıza çıkar. Programın çalışmasını etkilemezler.
- New sayesinde olayın farkında olduğunuzu derleyiciye söylemiş olursunuz. Derleyici de sizi uyarmayı bırakır.

Virtual (Sanal) Metotlar

- Virtual methods (sanal metotlar), base class (temel sınıf) içinde bildirilmiş ve derived class (türemiş sınıf) içinde de tekrar bildirilebilen metotlardır. Böylelikle sanal metotlar kullanılarak NYP'de çok sık başvurulan çok biçimliliği (polimorphism) uygulanmış olur. Yani temel sınıfta bir sanal metot bildirildiğinde bu temel sınıftan türeyen sınıflardaki metotlar override edilerek, temel sınıftaki sanal metodu devre dışı bırakabilirler.
- Sanal metotları bildirmek için virtual anahtar sözcüğü kullanılır. Bu anahtar sözcükle, sanal metotumuz bildirilmiş olur. Türeyen sınıfta ise, temel sınıftaki sanal metotları devre dışı bırakmak için override anahtar sözcüğü kullanılır. Yani base class'da virtual olarak tanımladığım bir metodu derived class içinde override edebilirim. Eğer override edersem derived class içindeki metot çalışır, eğer etmez isem direkt olarak base class içindeki virtual metot çalışır.
- Virtual metotlar private olarak tanımlanamazlar, public olmak zorundadırlar. Zaten private olmasının bir anlamı yok çünkü aksi halde derived class içinde bunları override edemeyiz. Public olmasının yanı sıra protected, internal şeklinde de bildirilebilirler.


Virtual Metotlar

```
public class Personel
{
    public string AdSoyad;
    protected decimal maas;
    public Personel(string _adsoyad, decimal _maas)
    {
        this.AdSoyad = _adsoyad;
        this.maas = _maas;
    }

    public virtual decimal UcretiHesapla()
    {
        return maas;
    }
}

public class Satıcı : Personel
{
    private decimal satis bonusu;
    public Satıcı(string _adsoyad, decimal _maas, decimal _satis bonusu)
        : base(_adsoyad, _maas)
    {
        this.satis bonusu = _satis bonusu;
    }

    public override decimal UcretiHesapla()
    {
        return maas + satis bonusu;
    }
}
```



Örneğimizi geçmiş konularla da birleştirdik.

Virtual bir metotta türemiş sınıf özgürdür. İster tanımlarsınız isterseniz tanımlamazsınız

Türemiş sınıfta tanımlama yaptığınızda override kullanımına dikkat.

ANA sınıfa virtual türetilmiş sınıfa override