Inheritance (Kalıtım, Miras alma)

- Inheritance (miras alma, kalıtım), bir nesnenin özelliklerinin farklı nesneler tarafından da kullanılabilmesine olanak sağlayan NYP özelliğidir.
- Öncelikle geçen haftaki konumuzda gördüğümüz:
  - Türetme
  - Erişim Belirteçleri
  - Yapıcılar ve Kalıtım

  konularını hatırlayalım.

Nesne Atama

- C# tür güvenliğine maksimum derecede önem vermektedir. Bu yüzden, eğer özel tür dönüşüm operatörleri bildirilmemişse farklı sınıf türlerinden nesneler birbirlerine atanamaz.

```
public class X
{
    public int a = 0;
}

public class Y
{
    public int b = 0;
}

class Program
{
    static void Main(string[] args)
    {
        Y _y = new Y();
        X _x = new X();

        _y = _x;
    }
}
```

Atama işlemi yapılamaz.

Ana ve yavru sınıf nesneleri

- Ancak türeyen sınıflarda bu kural delinir. Ana sınıf türünden nesnelere yavru sınıf türünden nesneler atanabilir. Örneğin C#'taki temel veri türlerinin object sınıfından türetildiğini söylemiştik. Bu sayede bir object nesnesine her türden veri atanabilir.

```
public class X
{
    public int a = 0;
}

public class Y : X
{
    public int b = 0;
}

class Program
{
    static void Main(string[] args)
    {
        Y _y = new Y();
        X _x = new X();

        _x = _y;
    }
}
```

Atanabilir, bu sayede base'in durumu değiştirilebilir.

Bu programda gözden kaçırmamamız gereken nokta nesne1 üzerinden B sınıfının kendine özgü üye elemanlarına erişemeyeceğimizdir. Bu şekilde ana sınıf türünden bir nesnenin kullanılabildiği her yerde yavru sınıf türünden bir nesneyi de kullanabiliriz. Örneğin bu örneğimizi düşünecek olursak bir metodun parametresi A tipinden ise bu metoda parametre olarak B tipinden bir nesneyi de verebiliriz. Çünkü her B nesnesi A nesnesinin taşıdığı bütün üye elemanları taşır. Bunu bilinçsiz tür dönüşümüne benzetebiliriz.

object sınıfı

- Şimdiye kadar bütün temel veri türlerinin object sınıfından türediğini söylemiştim. Asıl bombayı şimdi patlatıyorum. Bütün sınıflar gizlice object sınıfından türer. Bunu kanıtlamak için şu programı yazın:

```
using System;
class A
{
}
class Ana
{
   static void Main()
   {
      A nesne=new A();
      Console.WriteLine(nesne.ToString());
   }
}
```

- ToString() metodu normalde object sınıfına aittir. Ancak bütün sınıf nesneleriyle kullanılabilir. Bu programda ekrana nesnenin türü olan A yazılır. Başka bir örnek:

```
using System;
class A
{
}
class Ana
{
   static void Main()
   {
      A nesne=new A();
      object a=nesne;
   }
}
```

toString

```
public class X
{
    public override string ToString()
    {
        return "Ne istersen göster";
    }
}


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

- ToString() metodu normalde object sınıfına ait olduğunu söylemiştik. Eğer kendinize özel bir ToString() tanımlamak isterseniz override yapıp yeni bir string değeri döndürebilirsiniz.
- Override, (Geçersiz kılma) eski fonksiyonu geçersiz kılıp yeni durumu uygular.