

# Masaüstü Uygulama Geliştirme

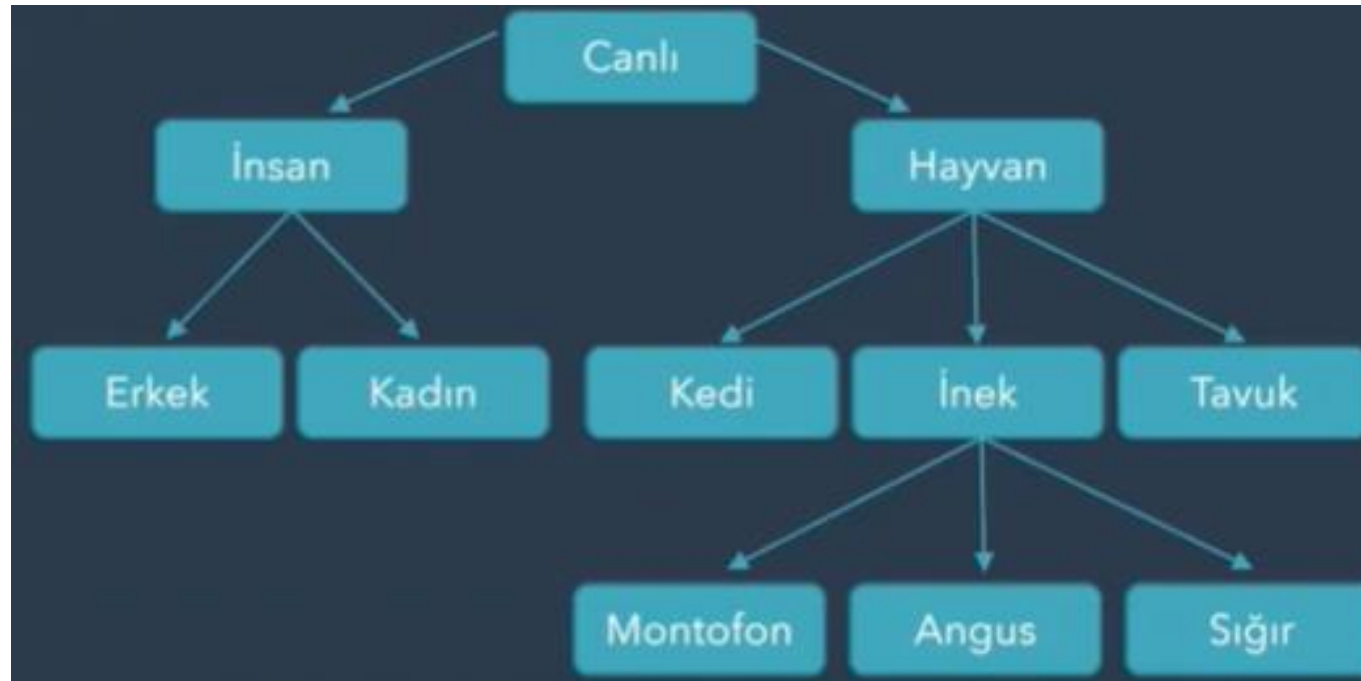
Hafta-12

Öğr. Gör. Zafer SERİN

# Kalıtım(Miras Alma - Inheritance)

- Kalıtım OOP'nin en önemli özelliğidir.
- Üretilen nesneler farklı nesnelere özelliklerini aktarabilmekte ve böylece hiyerarşik bir düzenleme yapılabilir.
- Bir programcı açısından bu özellik;
  - Aynı aile grubundan gelen nesnelerin ya da yatayda eşit seviyede olan tüm olguların benzer özelliklerini tekrar tekrar her birinde tanımlamaktansa bir üst sınıfta tanımlanmasını ve her bir sınıfın bu özellikleri üst sınıftan kalıtımsal olarak almasını sağlamaktadır.
  - Böylece kod maliyeti düşer ve mimarisel tasarım avantajı sağlanır.

# Kalıtım(Miras Alma - Inheritance)

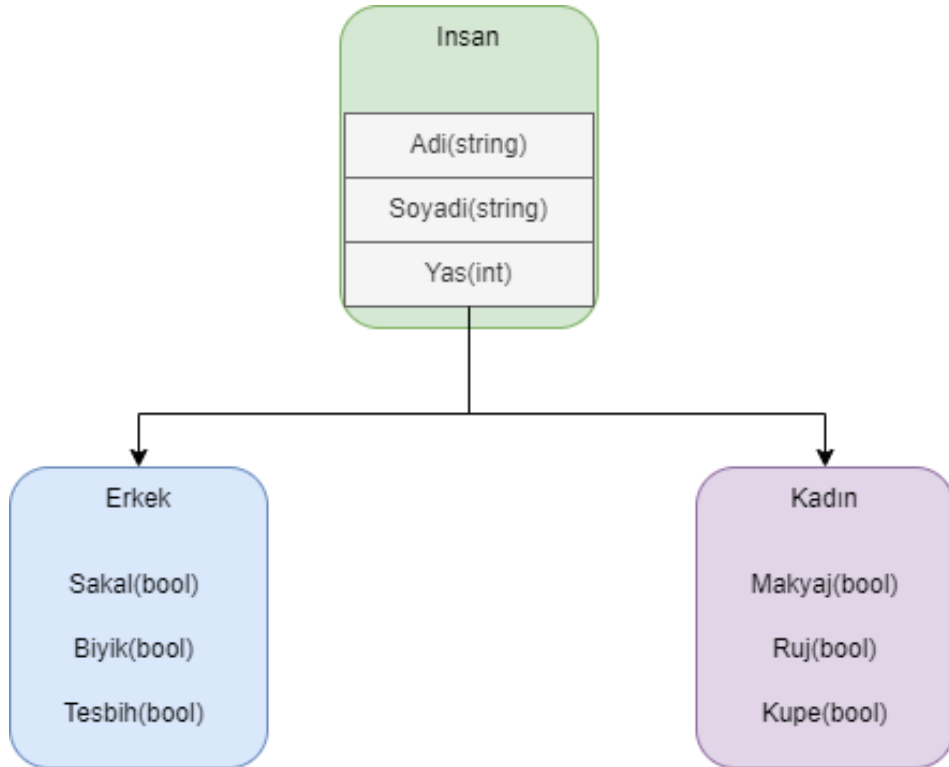


# Kalıtım(Miras Alma - Inheritance)

Erkek	Kadın
Adi(string)	Adi(string)
Soyadi(string)	Soyadi(string)
Yas(int)	Yas(int)
Sakal(bool)	Makyaj(bool)
Biyik(bool)	Ruj(bool)
Tesbih(bool)	Kupe(bool)

- Nesne tabanlı programlamada, benzer/aynı olgudaki nesnelerin aynı olan memberları/özellikleri/içerikleri eğer ki bu şekilde her sınıf içinde tekrar tekrar tanımlanmışsa bu aykırı bir durumdur!
- Aynı olguda olan sınıfların tekrar eden memberları başka bir sınıfta tanımlansın ve kalıtım ile diğer sınıflara aktarılsın.

# Kalıtım(Miras Alma - Inheritance)



- Erkek ve Kadın sınıfları İnsan sınıfından kalıtım alırsa/türetilirse/miras alırsa insan sınıfındaki tüm memberlar(erişimine izin verilen/miras olarak aktarılmasına izin verilen memberlar) miras olarak aktarılacaktır.
- Genellenemeyen, diğerlerinde olmayan ve sadece o sınıfa ait olan özellikler direkt ilgili sınıfta tanımlanmalıdır.

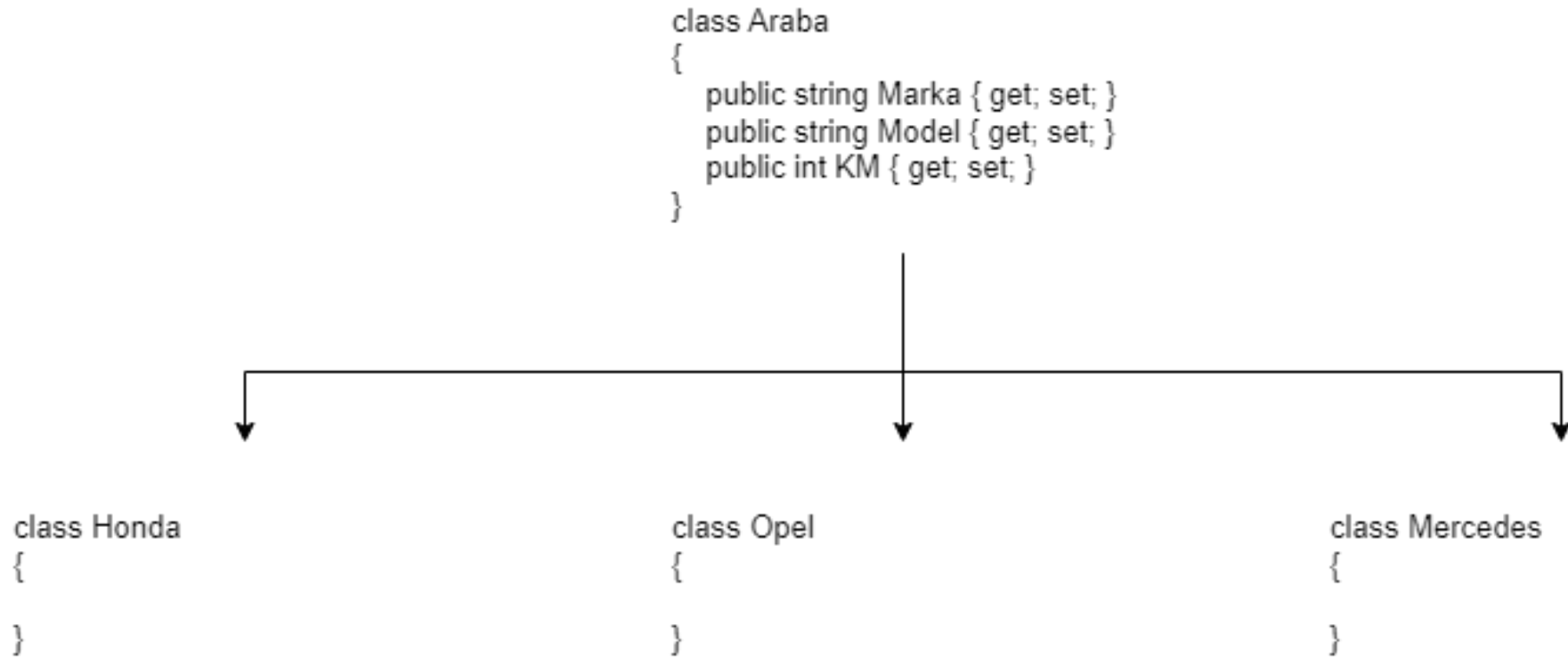
# Kalıtım(Miras Alma - Inheritance)

```
class Honda
{
    public string Marka { get; set; }
    public string Model { get; set; }
    public int KM { get; set; }
}
```

```
class Opel
{
    public string Marka { get; set; }
    public string Model { get; set; }
    public int KM { get; set; }
}
```

```
class Mercedes
{
    public string Marka { get; set; }
    public string Model { get; set; }
    public int KM { get; set; }
}
```

# Kalıtım(Miras Alma - Inheritance)



# Kalıtım(Miras Alma - Inheritance)

- Kalıtım rastgele bir şekilde tasarlanmamalıdır! Ortakolguda olan nesneleri temsil edecek olan bir üst ve daha evrensel nitelikte olgu olmalıdır. Honda, Opel ve Mercedes ortakolgudur. Yani üçünde bir arabadır. Haliyle bunların daha evrensel üst niteliği Araba olarak nitelendirilebilir.
- Kalıtım operasyonunda, kalıtım veren sınıfın erişilebilen tüm memberları kalıtım alan sınıfa kalıtsal olarak aktarılacaktır.



# Kalıtım(Miras Alma - Inheritance)

- OOP'de kalıtım özünde nesnelerin birbirlerinden türemesini sağlayan bir özelliktir.
- Bu özellik yanında da birçok özellik ve stratejik yapılanma getirmektedir.
- C#'ta iki sınıf arasında kalıtımsal ilişki kurabilmek için : operatörü kullanılmaktadır.

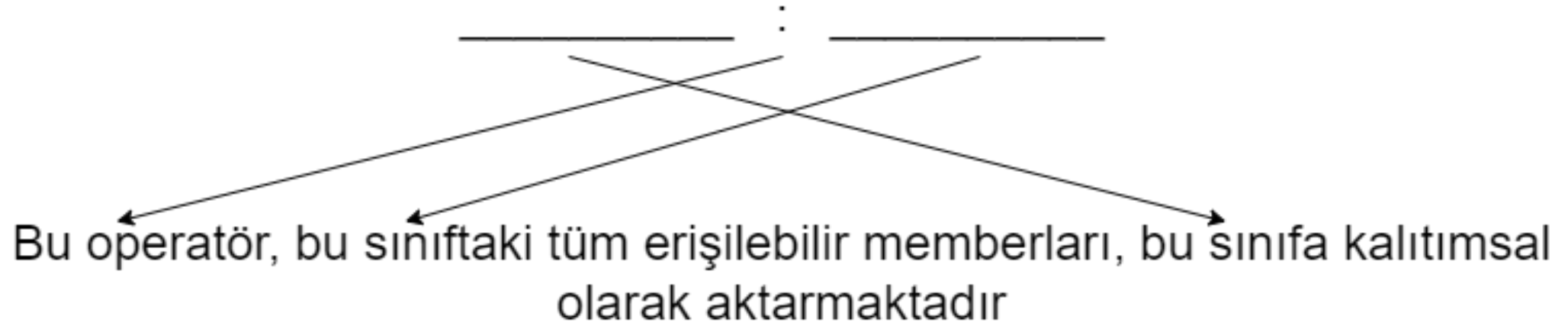
# Kalıtım(Miras Alma - Inheritance)

```
class Araba
{
    public string Marka { get; set; }
    public string Model { get; set; }
    public int KM { get; set; }
}
```

- : operatörünün solundaki class sağındaki classtan kalıtım yoluyla erişime açık olan memberları alacaktır.
- Yani; Honda sınıfı, Araba sınıfından kalıtım alsın demiş oluyoruz.

```
class Honda : Araba
{
}
```

# Kalıtım(Miras Alma - Inheritance)



# Kalıtım(Miras Alma - Inheritance)

- Kalıtım, operasyonel olarak gerçekleştirildikten sonra compiler seviyesinde member aktarımı sağlanır.
- Yani artık Honda sınıfından bir nesne ürettiğimizde içerisinde Marka, Model ve KM propertyleri kalıtımsal olarak aktarıldığı için erişilebilir olacaktır.

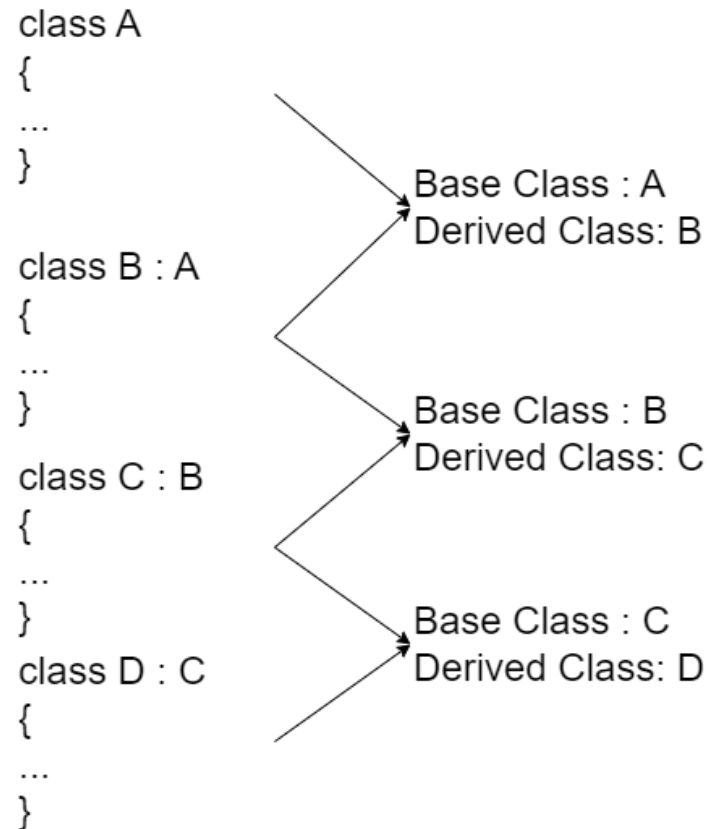
# Kalıtım(Miras Alma - Inheritance)

```
class Araba
{
    public string Marka { get; set; }
    public string Model { get; set; }
    public int KM { get; set; }
}
```

```
class Honda : Araba
{
}
```

- Kalıtım veren sınıfa Base / Parent Class denir.
- Kalıtım alan sınıfa Derived / Child Class denir.
- Bu örnek için Base Class Araba iken Derived Class Honda'dır.

# Kalıtım(Miras Alma - Inheritance)



- Peki, atalar tüm torunların Base Class'ı mıdır? Yani A, B'nin olduğu gibi bir yandan da C ve D'nin de Base Class'ı mıdır?
- Hayır değildir!
- Bir sınıfın sade ve sadece tek bir Base Class'ı olabilir.
- Yani bir sınıfın Base Class'ı direkt türediği sınıftır.
- Lakin atalarındaki tüm classlar Base Class'ı değildir.
- Örneğin; C'nin Base Class'ı B'dir. A ise atasıdır lakin Base Class'ı değildir.
- Buna karşın bir Class'ın birden fazla derived classı olabilir.

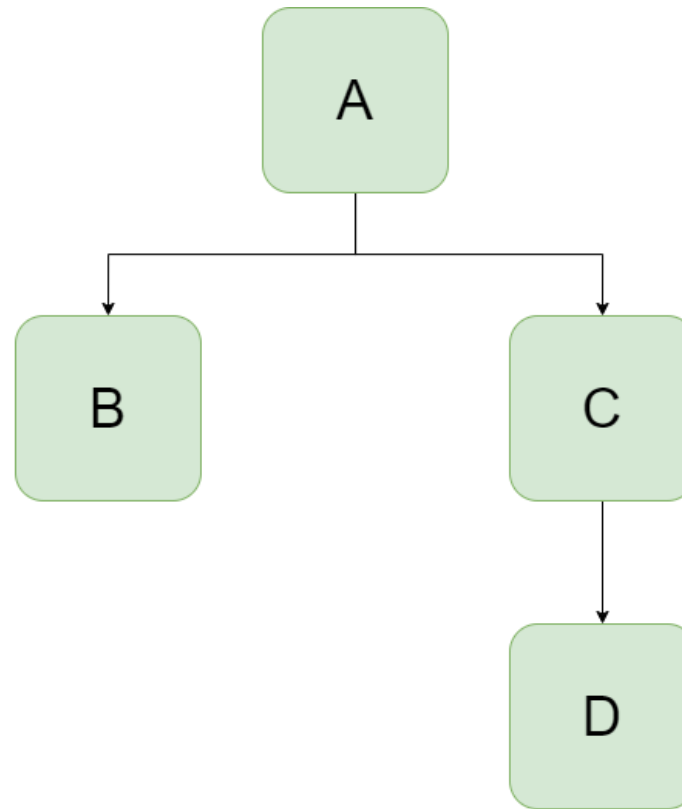
# Kalıtım(Miras Alma - Inheritance)

```
class A
{
...
}

class B : A
{
...
}

class C : A
{
...
}

class D : C
{
...
}
```



# Kalıtım(Miras Alma - Inheritance)

- Kalıtımın Altın Kuralı: Bir classın sade ve sadece bir Base Class'ı olabilir. Dolayısıyla bir classı hem A classından hem de B classından yatayda(aynı anda) türetmek mümkün değildir! Dikeyde bu işlem yapılabilir. Örneğin A classını aynı anda hem B'den hem de C'den türetemezsiniz, ancak B classını C'den türetilip A classında B'den türetebilirsiniz.
- Bunun nedeni, c# programlama dilinde bir classın sade ve sadece tek bir classtan türetilmesine izin verilmesidir! Aynı anda birden fazla classtan türeme işlemi gerçekleştirilemez!



# Kalıtım(Miras Alma - Inheritance)

- Bu durumda aşağıdaki gibi bir kullanım olamaz! (Not: bu durum classlar için geçerlidir. İleri düzey programlama yapılarından olan interface buna izin verecektir; ancak interface bir class değildir!)

```
class Y : X, Z, W  
{  
  
}
```

# Kalıtım(Miras Alma - Inheritance)

- Bir sınıftan nesne üretimi yapılırken kalıtım aldığı üst sınıflar varsa eğer önce o sınıflardan sırayla nesne üretilir. Örneğin new B() kodu ile aşağıdaki kod çalıştığında Heap'te önce A'dan sonra B'den nesne üretilir.

```
class A
{
}
class B : A
{
}
```

# Kalıtım(Miras Alma - Inheritance)

- Yani bir sınıftan nesne üretilirken 1 adet nesne üretildiği düşünülse de kalıtımsal açıdan birden fazla nesne üretimi gerçekleştirilebilmektedir.

# Kalıtım(Miras Alma - Inheritance)

- Herhangi bir sınıftan nesne üretimi gerçekleştirilirken öncelikle base classından nesne oluşturuluyor ise bu demektir ki önce base classın constructorı tetikleniyor.
- Haliyle bizler nesne üretimi esnasında base classta üretilecek olan nesnenin istediğimiz constructorlarını tetikleyebilmeli veya varsa parametrelerini verebilmeliyiz.
- İşte bu amaçlar için base keywordü kullanılır.

# Kalıtım(Miras Alma - Inheritance)

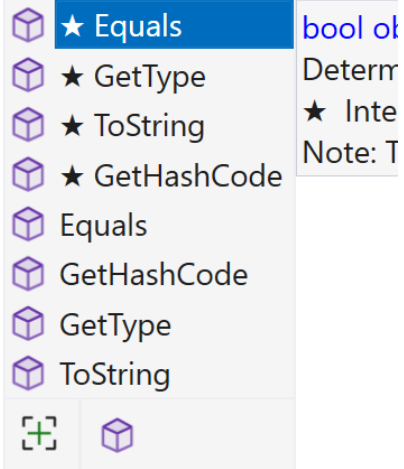
- Eğer ki base classın constructorı sadece parametre alan constructor ise derived classlarda o constructora bir değer göndermek zorundayız! Bunu da base keywordü ile sağlayabiliriz.
- Eğer ki base classta boş parametrelili bir constructor varsa derived classta base ile bir bildirimde bulunmak zorunda değiliz. Çünkü varsayılan olarak kalıtımsal durumda boş parametrelili constructor tetiklenir.
- Bir classın constructorının yanında :base() keywordü kullanılırsa; o classın base classının tüm constructorları bize getirilecektir. Bu sayede ilgili sınıftan bir nesne üretilirken base classın hangi constructorı tetiklenecek belirtebiliriz.

# Kalıtım(Miras Alma - Inheritance)

- this, bir sınıftaki constructorlar arasında geçiş yapmamızı sağlar.
- base, bir sınıfın base classının constructorlarından hangisinin tetikleneceğini belirlememizi ve varsa parametrelerinin değerlerinin derived classtan verilmesini sağlar.
- Buna ek olarak nasıl ki this, ilgili sınıfta o anki nesnenin memberlarına erişebilmemizi sağlıyor, aynı şekilde basede base classtaki memberlara erişebilmemizi sağlar.

# Kalıtım(Miras Alma - Inheritance)

```
namespace ConsoleApp1
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            İnsan insan1 = new İnsan();
            insan1.Equals(insan1);
        }
    }
    2 references
    class İnsan
    {
    }
}
```



- Nesnelerdeki ToString(), Equals(), GetHashCode() ve GetType() gibi metotlar c#'ta bütün nesnelerin atası olan Object sınıfından gelmektedir.

# Kalıtım(Miras Alma - Inheritance)

- C# programlama dilinde tüm sınıflar Object sınıfından türetilir. Bu işlem compiler seviyesinde gerçekleştirilir. Dolayısıyla bazı metotlar bu sayede alt sınıflara(torunlara) aktarılabilir.
- Dolayısıyla önceden gördüğümüz object yapısı Object classına ait bir yapı olduğu ve c#'taki tüm classlardan Object classından türetildiği için object anahtar sözcüğü ile her bir yapı tutulabilir, boxing ve unboxing işlemine tabi tutulabilir.



# Kalıtım(Miras Alma - Inheritance)

- Eğer ki tanımlanan sınıf herhangi bir kalıtım almıyorsa default(varsayılan) olarak Object sınıfından türetilcektir.
- Eğer ki tanımlanan sınıf herhangi bir sınıftan kalıtım alıyorsa, bir sınıfın aynı anda birden fazla sınıftan kalıtım almama prensibinden yola çıkarak bir yandan da Object sınıfından türemeyecek sadece kalıtım aldığı sınıftan türeyecektir.
- Tabi burada kalıtım veren sınıf herhangi bir sınıftan türemiyorsa eğer en nihayetinde Object'ten türeyeceği için dolaylı yoldan en alt sınıfta Object'ten kalıtım almış olacaktır.

# Eriřim Belirteçleri

Eriřim Belirteci	İřlevi
public	Eriřim kısıtı yoktur, her yerden erişilir.
protected	Ait olduđu sınıftan ve o sınıftan türetilen sınıflardan erişilebilir.
internal	Etkin projeye ait sınıflardan erişilebilir, onların dışından erişilemez.
private	Yalnız bulunduđu sınıftan erişilir, dıştaki sınıflardan erişilemez.
protected internal	Etkin projeye ait sınıflardan ve onların türevlerinden erişilebilir.
private protected	Yalnız bulunduđu sınıftan ve onların türevlerinden erişilebilir