Nested Structs (İç içe yapılar)

- Sınıf ve yapılar iç içe kullanılabilir.

```
//iç içe yapılar (nested structure)
public struct S1
{
    public string A;

    public struct S2
    {
        public string B;
    }
}

//iç içe sınıflar (nested class)
public class C1
{
    public string A;

    public class C2
    {
        public string B;
    }
}
```

```
class Program
{
    static void Main(string[] args)
    {
        /*S1 _s1 = new S1();
        _s1.A = "Değer 1";

        S1.S2 _s12 = new S1.S2();
        _s12.B = "İç Değer";*/

        //new kullanmadan
        S1 _sn1;
        _sn1.A = "Değer 1";

        S1.S2 _sn12;
        _sn12.B = "İç Değer";

        //sınıf yaratılmak zorundadır.
        C1 _c1 = new C1();
        C1.C2 _c2 = new C1.C2();
    }
}
```

Hem class hem de struct'ta B'ye sırasıyla C2 ve S2 üzerinden ulaşılabilir.

Struct vs. Class

- Class'la referans tipindendir (reference types)
- Struct'lar ise de değer tipindendir. (value types)
- Yapılar kalıtım / miras desteklemez.
- Yapıların geçerli constructor'a sahip değildir. Diğer deyişle constructor'ları parametre içermek zorundadır.

Struct vs. Class

```
public class sinif
{
    public int x;
}

public struct yapi
{
    public int x;
}
```

- Bu örnekte referans tipinden olan sınıf ile değer tipinden olan yapı yaratılacak ve aktarımlar üzerinden fark anlatılacaktır.

Struct vs. Class

```
public class sinif
{
    public int x;
}

public struct yapi
{
    public int x;
}
```

```
static void Main(string[] args)
{
    sinif _c1 = new sinif();
    _c1.x = 5;

    sinif _c2 = new sinif();
    _c2 = _c1;//bellek üzerine işlem yapılır. Class referans türüdür.

    Console.WriteLine("Class ve Nesne üzerine olan işlemler");
    _c1.x = 8;
    Console.WriteLine(_c2.x.ToString());

    _c2.x = 7;
    _c1.x = 10;
    Console.WriteLine(_c1.x.ToString());
    Console.WriteLine(_c2.x.ToString());

    Console.WriteLine("Struct üzerine olan işlemler");
    yapi _s1, _s2;
    _s1.x = 5;
    _s2 = _s1;
    Console.WriteLine(_s2.x.ToString());

    //Struct değer türüdür. bellek adresleri üzerine değil, direkt bellekteki içerik üzerine işlem yapılır.
    _s1.x = 13;
    _s2.x = 1;
    Console.WriteLine(_s1.x.ToString());
    Console.WriteLine(_s2.x.ToString());

    Console.ReadLine();
}
```

Her iki çıktı sonucu: 10
_c1.x'teki değişim her iki sınıfı etkiler.

Çıktılar sırasıyla: 13 ve 1'dir.
_s1.x'teki değişim sadece kendi yapısını etkiler.

- _c2=_c1 işleminde adres ataması yapılır. Bu aşamadan sonra _c1 üzerindeki her değişiklik _c2'ye yansır. Başka _c1 ve _c2 adresleri eşitlenmiştir.
- _s2 = _s1 işleminde ise sadece değer ataması yapılmıştır. Her iki değişkenin çalışma adresi halen birbirinden farklıdır.