# Bölüm 24

# Indeksci (indexer)



## İndeksci Nedir?

İndeksci bir sınıfa ait nesneleri sanki bir arrayin bileşenleri imiş gibi indeksler. Sınıfa ait öğelerin ne olduğu indeksci için önemli değildir. O yalnızca yaratılan nesnelere damga (indis) koyar. Sınıfa ait `özgen` (property) tanımlamaya çok benzer. O nedenle, ona *akıllı array* takma adı verilmiştir. İndeksçiler *private, public, protected ya da internal* erişim belirteçleriyle nitelenebilirler. Aşağıdaki program nesneleri tamsayı indisler veren basit bir indeksçidir.

Indeksci01.cs

```
using System;

    class IndesksYap
    {
        private string[] metin = new string[5];
        public string this[int i]
        {
            get
            {
                return metin[i];
            }
            set
            {
                metin[i] = value;
```

```
            }
        }
    }

    class NesneYap
    {
        public static void Main()
        {
            IndesksYap obj = new IndesksYap();
            obj[0] = "Konya";
            obj[1] = "Elma";
            obj[2] = "Şiir";
            obj[3] = "Okul ";
            obj[4] = "C# ile Nesne Programlama";
            for (int j = 0; j < 5; j++)
            {
                Console.WriteLine("{0}", obj[j]);
            }
        }
    }
}
```

Çıktı

```
Konya
Elma
Şiir
Okul
C# ile Nesne Programlama
```

Bu programı çözümleyerek indekscilerin nasıl çalıştığını açıklayabiliriz. `İndeksYap` sınıfındaki

```
private string[] metin = new string[5];
```

deyimi string türünden, metin adlı, 5 bileşenli, `private` nitelemeli bir `array` bildirimidir.

```
        public string this[int i]
        {
            get
            {
                return metin[i];
            }
            set
            {
                metin[i] = value;
            }
        }
```

blokunda `this` pointeri yaratılan arrayi işaret eder. `get/set` erişimcileri (accessors), arrayin bileşenlerine değer atar ve atanan değeri okur.

`NesneYap` sınıfı içindeki `Main()` metodu

```
IndesksYap obj = new IndesksYap();
```

deyimiyle `IndeksYap` sınıfına ait obj adlı bir nesne yaratır. Bu nesnenin string tipinden, 5 bileşenli bir array olduğunu biliyoruz. `Main()` metodunun sonraki kodları arrayin bileşenlerine değerler atıyor sonra o değerleri *for döngüsü* ile konsola yazdırıyor.

Aşağıdaki örnek `double` tipinden sayıları bir `array` gibi kullanmamızı sağlar.

Indeksci02.cs

```
using System;

class IntIndexer
{
    private double[] kesir;

    public IntIndexer(int boyu)
    {
        kesir = new double[boyu];

        for (int i = 0; i < boyu; i++)
        {
            kesir[i] = 100.0;
        }
    }

    public double this[int ndx]
    {
        get
        {
            return kesir[ndx];
        }
        set
        {
            kesir[ndx] = value;
        }
    }

    static void Main(string[] args)
    {
        int boyu = 10;

        IntIndexer aaa = new IntIndexer(boyu);

        aaa[7] = 12.8;
        aaa[2] = 5.3;
        aaa[4] = 6.0;

        for (int i = 0; i < boyu; i++)
        {
            Console.WriteLine("aaa[{0}] = {1} ", i, aaa[i]);
        }
    }
}
```

Çıktı

```
aaa[0] = 100
aaa[1] = 100
aaa[2] = 5,3
aaa[3] = 100
```

```
aaa[4] = 6
aaa[5] = 100
aaa[6] = 100
aaa[7] = 12,8
aaa[8] = 100
aaa[9] = 100
```

Programı çözümleyelim. `IntIndexer` sınıfındaki

```
private double[] kesir;
```

deyimi double tipinden kesir adlı ve private nitelemeli bir array yaratır. private nitelemeli olduğu için bu arraye sınıf dışından erişilemez. O nedenle erişimcileri (accessors) kullanıyoruz.

```
public IntIndexer(int boyu)
{
    kesir = new double[boyu];

    for (int i = 0; i < boyu; i++)
    {
        kesir[i] = 100.0;
    }
}
```

bloku `IntIndexer` sınıfının 1 parametreli bir kurucusudur. `kesir` arrayinin bütün bileşenlerine 100.0 değerini atıyor. Bu değer atanmasa da olur. O zaman `double` tipi bileşenler *öndeğer* (default value) olarak 0.0 değerini alırlar.

```
public double this[int ndx]
{
    get
    {
        return kesir[ndx];
    }
    set
    {
        kesir[ndx] = value;
    }
}
```

blokunda `this` pointeri kesir arrayini gösteriyor. Bu pointer yardımıyla `get/set` erişimcileri (accessor methods) arrayin bileşenlerine değer atıyor ve atanan değeri okuyor.

`Main()` metodu, ilkönce arrayin uzunluğunu `boyu = 10` deyi ile belirliyor. Sonra

```
IntIndexer aaa = new IntIndexer(boyu);
```

deyimi ile IntIndexer sınıfına ait aaa nesnesini yaratıyor. aaa içindeki kurucu, 10 bileşenli kesir adlı arrayi oluşturuyor. set metodu bütün bileşenlere 100.0 değerini atamıştı.

```
aaa[7] = 12.8;
aaa[2] = 5.3;
aaa[4] = 6.0;
```

deyimleri 7, 2 ve 4 damgalı bileşenlerin değerlerini değiştiriyor, yeni değerler atıyor. Bu kodlar, arrayin bileşenlerine atanan değerlerin istendiğinde değiştirilebileceğini gösteriyor.

`Main()` metodunun içindeki son blok for döngüsüyle arrayin bileşenlerinin değerlerini konsola yazdırıyor.

## Aşkın İndeksci

İndeksçiler birer metot olduğuna göre, aşkın indeksçi olması doğaldır. Aşağıdaki program bir aşkın indeksçi (overloaded indexer) tanımlıyor.

Indeksci03.cs

```
using System;

class AIndexer
{
    private string[] kent;
    private int arrBoy;

    public AIndexer(int boy)
    {
        arrBoy = boy;
        kent = new string[boy];

        for (int i = 0; i < boy; i++)
        {
            kent[i] = "***";
        }
    }

    public string this[int ndx]
    {
        get
        {
            return kent[ndx];
        }
        set
        {
            kent[ndx] = value;
        }
    }

    public string this[string veri]
    {
        get
        {
            int sayaç = 0;

            for (int i = 0; i < arrBoy; i++)
            {
                if (kent[i] == veri)
                {
                    sayaç++;
                }
            }
            return sayaç.ToString();
        }
        set
        {
            for (int i = 0; i < arrBoy; i++)
```

```
            {
                if (kent[i] == veri)
                {
                    kent[i] = value;
                }
            }
        }
    }

    static void Main(string[] args)
    {
        int boy = 10;
        AIndexer aaa = new AIndexer(boy);

        aaa[9] = "Rize";
        aaa[3] = "Trabzon";
        aaa[5] = "Sinop";
        for (int i = 0; i < boy; i++)
        {
            Console.WriteLine("aaa[{0}]  =  {1}", i, aaa[i]);
        }
    }
}
```

Çıktı

```
aaa[0] = ***
aaa[1] = ***
aaa[2] = ***
aaa[3] = Trabzon
aaa[4] = ***
aaa[5] = Sinop
aaa[6] = ***
aaa[7] = ***
aaa[8] = ***
aaa[9] = Rize
```

Bu programı çözümleyelim. Aindexer sınıfındaki

```
private string[] kent;
```

deyimi private nitelemeli, string tipinden kent adlı bir array bildirimidir.

```
private int arrBoy;
```

deyimi, kent arrayinin bileşen sayısını tutacak bir değişken bildirimidir.

```
    public AIndexer(int boy)
    {
        arrBoy = boy;
        kent = new string[boy];

        for (int i = 0; i < boy; i++)
        {
            kent[i] = "***";
        }
    }
```

bloku `AIndexer` sınıfının int tipinden 1 parametreli bir kurucusudur. String tipinden kent arrayini yaratır ve bileşenlerine "***" değerini atar.

```
    public string this[string veri]
    {
        get
        {
            int sayaç = 0;

            for (int i = 0; i < arrBoy; i++)
            {
                if (kent[i] == veri)
                {
                    sayaç++;
                }
            }
            return sayaç.ToString();
        }
        set
        {
            for (int i = 0; i < arrBoy; i++)
            {
                if (kent[i] == veri)
                {
                    kent[i] = value;
                }
            }
        }
    }
```

blokunda this pointeri arrayi işaret eder. Dolayısıyla, ilk satır [string veri] parametresi ilk kurucunun int tipi parametre tipinden farklıdır. O halde, bu bir aşkın kurucudur. Bu bloktaki get/set erişimcileri (accessors) `kent` arrayinin bileşenlerine for döngüleriyle değer atıyor ve o değerleri okuyor.

`Main()` metodu arrayin bileşen sayısını 10 olarak belirledikten sonra `Aindexer` sınıfının `aaa` adlı bir nesnesini yaratıyor. Sonra arrayin 9, 3, 5 damgalı bileşenlerine değer atıyor ve bütün bileşen değerlerini for döngüsü ile konsola yazdırıyor.

## Alıştırma

Aşağıdaki programın deyimlerini çıktı ile karşılaştırarak çözümleyiniz.

### Indeksci04.cs

```
using System;
//using System.Collections.Generic;
//using System.Text;

namespace Indexers
{
    class Ata
    {
        private string[] bölge = new string[5];

        public string this[int indexbölge]
```

```
        {
            get
            {
                return bölge[indexbölge];
            }

            set
            {
                bölge[indexbölge] = value;
            }
        }
    }

    /* Ata sınıfının nesnelerini bir array gibi kullanır */
    class Oğul
    {
        public static void Main()
        {
            Ata obj = new Ata();

            obj[0] = "Matematik";
            obj[1] = "Fizik";
            obj[2] = "Kimya";
            obj[3] = "Biyoloji";
            obj[4] = "İstatistik";

            Console.WriteLine("{0}\n,{1}\n,{2}\n,{3}\n,{4}\n", obj[0],
obj[1], obj[2], obj[3], obj[4]);
        }
    }
}
```

Çıktı

```
Matematik
,Fizik
,Kimya
,Biyoloji
,İstatistik
```