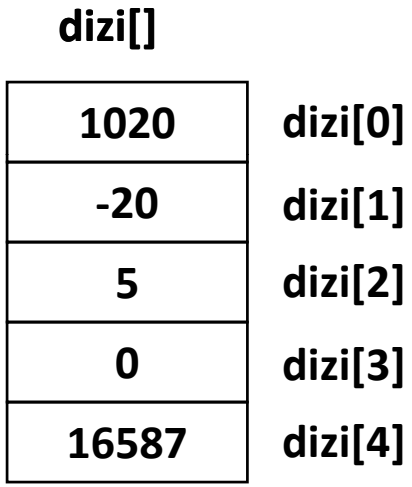
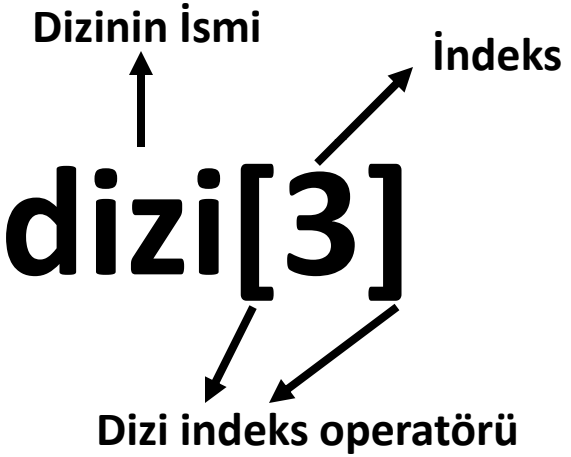
Diziler (Arrays)

Dizi Kavramı

- Bellekte ard arda yer alan aynı türden nesneler kümesine dizi (array) denilir. Bir dizi içerisindeki bütün elemanlara aynı isimle ulaşılır. Yani dizideki bütün elemanların isimleri ortaktır. Elemanlar arasındaki ayırt edici özellik, bellekteki yeridir.

5 elemanlı bir tamsayı dizisi bellekte aşağıdaki gibi yerleştirilir.





- C#'da diziler ayrı bir tür olarak tasarlanmıştır. Bütün diziler System.Array sınıfından türetilmiştir. Bir dizi tanımlandığında Array sınıfı türünden bir nesne tanımlanmış olur.

- Dizinin sınırlarını aşan bir indeks kullandığımızda derleme hatası oluşmaz, ancak çalışma zamanında hata oluşur.

Dizi Tanımlama

2 türlü tanımlama yapılabilir :

- ```
  int [ ]  dizi = new int[19]
  ```

veya

- ```
  int [ ] dizi;
  dizi=new int [19];
  ```

- new anahtar sözcüğü ile, dizinin her elemanına temel veri türleri için varsayılan değer, ilk değer olarak verilmektedir. Bu ilk değer; referans türleri için null, nümerik türler için 0, bool türü için ise false'tur.

- Dizileri aşağıdaki gibi de tanımlamak mümkündür:

```
string[ ] dizi1={“Bilgisayar”,“Mühendisliği”,“Bölümü”};
int [ ]={5,9,12,56,23};
float [ ] dizi3={8f,39f,324f,23f,2f};
```

- Dizilerin boyutu C ve C++ dillerinde derleme sırasında bilinmek zorundadır. Böylece derleyici, dizi için bellekte dizinin boyutuna göre yer ayırmaktadır. Ancak C#'da dizilerin boyutlarını çalışma zamanında belirtmek mümkündür.

Aşağıdaki kod parçasını bilgisayarınızda deneyiniz:

```
class Program
    {
        static void Main(string[] args)
        {
            Console.Write("Dizinin boyutunu giriniz=");
            int boyut = Convert.ToInt32(Console.ReadLine());
            bool [] dizi = new bool [boyut];
            for (int i = 0; i < boyut; i++)
                Console.WriteLine(dizi[i]);
            Console.ReadLine();
        }
    }
```

Bir dizinin boyutu bir kez belirlendikten sonra artık değiştirilemez. Yani dizinin boyutunu dinamik olarak değiştirmemiz mümkün değildir.

System.Random Sınıfı

Rastgele sayı üretmek için .NET Framework sınıf kütüphanesindeki System.Random sınıfı kullanılır. Rastgele sayı üretmek için öncelikle Random sınıfı türünden bir nesne oluşturmak gerekir. Örn:

```
Random rnd = new Random();
```

rnd nesnesi oluşturulduktan sonra artık bu nesnenin metotlarına erişebiliriz. Örn:

```
int rastgelesayi=rnd.Next (10,20);

int rastgelesayi=rnd.Next(50);

int rastgelesayi=rnd.Next;

double rastgelesayi=rnd.NextDouble();
```

```
using System;

class dizi_3
{
    static void Main()
    {
        Random r = new Random();

        int[] dizi = new int[20];
        char[] chr = new char[20];

        for (int i = 0; i < 20; i++)
        {
            dizi[i] = r.Next(1, 51);
            chr[i] = (char)r.Next(20, 126);
        }
        for (int i = 0; i < 20; i++)
        {
            Console.Write("{0,2}. değer {1,2} -> ", i, dizi[i]);
            for (int j = 0; j < dizi[i]; j++)
                Console.Write(chr[i]);
            Console.WriteLine();
        }
    }
}
```

Çok Boyutlu Diziler (Multidimensional Arrays)

- İki türlü çok boyutlu dizi bulunur. Her boyutta eşit miktarda elemanın olduğu düzenli diziler (matrisler) ve her boyutta farklı sayıda eleman barındıran düzensiz diziler(jagged arrays). Aslında düzensiz diziler elemanları da dizi olan dizilerdir.

- Çok boyutlu dizi tanımlaması yaparken "[, ,]" şeklinde yazılır.

Matris Dizileri

```
int [,] dizi = {{1,2},{3,4},{5,6}};
```

Şeklinde tanımlanan bir 3x2'lik dizinin elemanları:

dizi[0,0]=1
dizi[0,1]=2
dizi[1,0]=3
dizi[1,1]=4
dizi[2,0]=5
dizi[2,1]=6

dizi

[0,0]	[0,1]
[1,0]	[1,1]
[2,0]	[2,1]

Aşağıdaki programı bilgisayarınızda deneyiniz.

```
using System;

class dizi_4
{
    static void Main()
    {
        int[,] dizi = { { 1, 2 }, { 3, 4 }, { 5, 6 } };

        for (int i=0; i<3; i++)
            for (int j =0; j < 2; j++)
            {
                Console.WriteLine(dizi[i, j]);
            }
    }
}
```

Düzensiz Diziler(Jagged Arrays)

Dizilerin elemanları da farklı diziler olabilir. Bu durumda tanımlanan dizinin birbirinden farklı boyutlarda alt elemanları olabilir. Bu gibi yapılara düzensiz diziler ismi verilir.

Bu diziler şu şekilde tanımlanırlar:

```
int[][] dizi new int[3][];
dizi[0] = new int[2];
dizi[1] = new int[4];
dizi[2] = new int[3];
```

- Düzensiz tanımlanan bir dizinin yapısı şu şekildedir:

dizi

dizi[0]	**dizi[0][0]**	**dizi[0][1]**		
dizi[1]	**dizi[1][0]**	**dizi[1][1]**	**dizi[1][2]**	**dizi[1][3]**
dizi[2]	**dizi[2][0]**	**dizi[2][1]**	**dizi[2][2]**	

- Düzensiz dizilerde eleman sayısı sabit olmadığı için dizi elemanlarını taramak için Array sınıfının metotları ve ya özellikleri kullanılır. Length özelliği bir dizinin eleman sayısını verir.

```
using System;

class dizi_5
{
    static void Main()
    {
        int[][] dizi = new int[3][];

        dizi[0] = new int[] { 1, 2 };
        dizi[1] = new int[] { 3, 4, 5, 6, 7 };
        dizi[2] = new int[] { 8, 9, 0 };

        for (int i = 0; i < dizi.GetLength(0); i++)
         for (int j = 0; j < dizi[i].GetLength(0); j++)
         Console.WriteLine("dizi[{0}][{1}]={2}",i,j,dizi[i][j]);
    }
}
```

```
using System;

class dizi_6
{
    static void Main()
    {
        int[][] dizi = new int [3][];

        dizi[0] = new int[] {1,2};
        dizi[1] = new int[] {3,4,5,6,7};
        dizi[2] = new int[] {8,9,0};

        foreach (int[] boyut in dizi)
        {
            foreach (int eleman in boyut)
            {
                Console.Write("{0,3}", eleman);
            }
            Console.WriteLine();
        }
    }
}
```

System.Array Sınıfı

- Dizi nesneleri System.Array sınıfından türemişlerdir. Dolayısıyla bu sınıfın bazı özellik ve metotlarını bünyelerinde barındırırlar.

Özellikler:

IsFixedSize	Dizinin eleman sayısının sabit olup olmadığını verir. (Boolean)
IsReadOnly	Dizideki elemanların sadece okunabilir olup olmadığını verir. (Boolean)
Length	Dizideki eleman sayısını verir.
Rank	Dizinin boyutunu verir.

Metotlar:

BinarSearch	Tek boyutlu dizide binary search algoritmasına göre arama yapar.
Clear	Dizinin elemanlarını varsayılan değere çeker.
Clone	Dizinin bit bit kopyasını çıkarır.
Copy	Dizinin bir bölümünü başka bir diziye kopyalar.
CopyTo	Bir dizinin belirlenen bir kısmını başka bir diziye kopyalar.
GetLength	Dizideki eleman sayısını verir.
GetValue	Dizideki ilgili eleman değerini verir.
IndexOf	Dizi içindeki bir değerin ilk görüldüğü indeksi verir.
Reverse	Diziyi tersine çevirir.
SetValue	Bir dizinin bir elemanına değer atar.
Sort	Bir boyutlu dizilerde sıralama yapar.
CreateInstance	Yeni bir dizi nesnesi oluşturur.

CreateInstance( )

- CreateInstance metodu ile aşağıdaki gibi dizi nesnesi oluşturulabilir:

```
Array dizi=Array.CreateInstance(typeof(int),5);
```

- İlk parametre her zaman Type türünden olmalıdır. Yukarıdaki gibi bir kullanımda, typeof() operatörü yardımıyla int türünün type sınıfındaki karşılığı elde edilmiş olmaktadır. Örn:

```
CreateInstance(Type,int,int,int);
```

3 boyutlu bir dizi oluşturulmakta.

CreateInstance( )

```
using System;

class dizi_6
{
    static void Main()
    {
        Array dizi = Array.CreateInstance(typeof(int), 5, 4, 3);
        Random r = new Random();

        for (int i=0; i< dizi.GetLength(0); i++)
            for (int j=0; j< dizi.GetLength(1); j++)
                for (int k = 0; k < dizi.GetLength(2); k++)
                {
                    dizi.SetValue(r.Next(10, 100), i, j, k);
                    Console.WriteLine("dizi[{0},{1},{2}]={3,3}",
                                      i,j,k,dizi.GetValue(i,j,k));
                }
    }
}
```

CopyTo ( )

CopyTo metodu ile bir dizinin tamamı, başka bir dizinin istenilen yerine kopyalanabilir. Örn:

```
int[ ] dizi1= {1,2,3,4};
int  dizi2=new int[10];
dizi1.CopyTo(dizi2,3);
```

Aynı işlem Copy() metodu kullanılarak da gerçekleştirilebilir.

```
Copy(Array dizi1,Array dizi2,int uzunluk);
```

Bu şekildeki bir kullanım, uzunluk kadar elemanı dizi1'den dizi2'ye kopyalar. Kopyalama işlemi 0. indekten başlar.

```
Copy(Array dizi1,int x,Array dizi2,int y,int uzunluk);
```

dizi1'in x. İndeksinden sonraki uzunluk kadar elemanı dizi2'nin y. İndeksinden sonrasına kopyalar.

CopyTo ( )

```
using System;

class dizi_7
{
    static void Main()
    {
        int[] dizi1 = { 1, 2, 3, 4, 5, 6, 7 };
        int[] dizi2 = new int[10];
        int[] dizi3 = new int[10];

        dizi1.CopyTo(dizi2, 2);
        foreach (int i in dizi2)
            Console.Write(i);

        Console.WriteLine();

        Array.Copy(dizi1, 2, dizi3, 5, 3);

        foreach (int i in dizi3)
            Console.Write(i);
    }
}
```

Array.Sort ( )

Sort() metodunun birçok kullanım biçimi vardır. Temelde aşağıdaki şekilde kullanımı yaygındır.Örn:

```
Array.Sort(Array);
```

Array.Sort ( )

```
using System;

class dizi_8
{
    static void Main()
    {
        string[] dizi = { "Zeynep", "Fatma", "Ali", "Yılmaz",
   "Gökhan", "Osman", "Feride" };

        Console.WriteLine("Dizinin Elemanları\n==================");
        foreach (string s in dizi)
            Console.WriteLine(s);

        Array.Sort(dizi);
        Console.WriteLine();

        Console.WriteLine("Sıralanmış Dizi\n===============");
        foreach (string s in dizi)
            Console.WriteLine(s);

    }
}
```

Array.BinarySearch

BinarySearch metodunun iki önemli kullanım biçimi vardır
Örn:

- BinarySearch(Array dizi, object nesne);

Bu yapı, dizi içerisinde nesneyi arar; eğer aranan nesne bulunursa bulunduğu index ile geri dönülür. Eğer bulunamazsa negatif bir sayıyla geri dönülür.

- BinarySearch(Array dizi, int baslangic, int uzunluk, object nesne);

Bu yapı ise, nesneyi başlangic indeksinden itibaren uzunluk kadar eleman içerisinde arar.

Array.BinarySearch

```
using System;

class dizi_9
{
    static void Main()
    {
        string[] dizi = { "Zeynep", "Fatma", "Ali", "Yılmaz",
   "Gökhan", "Osman", "Feride" };

        Console.Write("Aranacak İsmi Giriniz :");
        string isim = Console.ReadLine();

        Array.Sort(dizi);
        int indeks = Array.BinarySearch(dizi, isim);

        if (indeks < 0)
            Console.WriteLine("Aranan isim dizide bulunamadı!");
        else
            Console.WriteLine("Aranan isim dizinin {0}. elemanında
   bulundu...",indeks);
    }
}
```

Diğer İşlemler

- Clear() metodu, belirtilen indis aralığındaki elemanları sıfırlar.

```
Array.Clear(dizi,2,5);
```

- Reverse() metodu, dizinin tümünü yada belirtilen indis aralığındaki elemanları ters çevirir.

```
Array.Reverse(dizi); veya
Array.Reverse (dizi,1,3);
```

Alttaki gibi bir kullanımda ise,1. elemandan itibaren ilk 3 elemanı ters çevirir.