C Sharp Programlama Dili/Diziler

Ders 9. Diziler

Şimdiye kadar birçok değişken tanımladık ve programlarımızın içinde kullandık. Bir program içinde tanımladığımız değişken sayısı şimdiye kadar bir elin parmaklarını geçmedi. Ancak her zaman bu böyle olmayabilir. Bazı programlarda 200-300 değişkene ihtiyaç duyabiliriz. Bunların hepsinin teker teker tanımlanması oldukça zahmetlidir. İşte bu yüzden programlama dillerinde dizi diye bir kavram vardır. Aslında bir dizi, birbiriyle ilişkili değişkenlerin oluşturduğu bir gruptan başka bir şey değildir. Diyelim ki `sayi1` ve `sayi2` adlı iki değişken tanımladık. Bunların adları birbirine benzemesine rağmen birbiriyle hiçbir ilişkisi yoktur. Yani bir döngü içinde `sayiX` yazarak bu değişkenlere erişemeyiz. Halbuki dizilerde bu mümkündür.

Dizilerin tanımlanması ve elemanlarının kullanılması

```
int[] dizi=new int[25];
```

veya

```
int[] dizi;
dizi=new int[25];
```

Yukarıdaki iki kodda da int türünden 25 elemanlı `dizi` adında bir dizi tanımlandı ve dizinin her bir elemanına int türünün varsayılan değeri atandı. Varsayılan değerler, sayısal türler için 0, object türü için NULL (yokluk), string türü için "", char için ' ' (boşluk) ve bool için false değerleridir.

Bütün dizilerin birinci elemanı 0. indeksidir. `dizi` dizisinin birinci elemanına `dizi[0]`, 10. elemanına `dizi[9]` yazarak erişebilir ve bu dizi elemanlarını bir değişkenmiş gibi kullanabiliriz. Örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      int[] dizi=new int[20];
      dizi[5]=30;
      Console.Write(dizi[5]);
   }
}
```

Bu program ekrana 30 yazacaktır.

C ve C++ programlama dillerinde olduğu gibi dizilerin elemanlarına aşağıdaki gibi de değer atayabiliriz:

```
string[] dizi1={"Bir","İki","Üç"};
int[] dizi2={2,-4,6};
float[] dizi3={2f,1.2f,7f};
```

Ancak bu şekilde dizi belirtimini sadece dizi tanımlamalarında kullanabiliriz. Örneğin bir sonraki derste göreceğimiz metotlara parametre olarak bir int dizisi vermemiz gerekiyorsa parametre olarak sadece `{1, 2, 3}` ifadesini veremeyiz. `{1, 2, 3}` ifadesini bir değişkene atayıp bu değişkeni metoda parametre olarak vermeliyiz. Ayrıca aşağıdaki kullanım da hatalıdır:

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

Çünkü daha önce de bahsettiğimiz gibi direkt dizi belirtimi sadece dizi tanımlamalarında geçerlidir.

Başka bir dizi tanımlama yöntemi de şöyledir:

```
int[] dizi=new int[]{1, 2, 3};
```

Üstelik bu şekilde dizi belirtimini dizi kullanmamız gereken her yerde yapabiliriz. Örneğin bir metoda parametre olarak bir int dizisi vermemiz gerekiyorsa metot çağrısında parametre yerine direkt `new int[]{1, 2, 3}` yazabiliriz.

Diziler yukarıdaki şekilde tanımlandığında söz konusu dizilerin eleman sayısı yazılan eleman sayısı olur. Örneğin yukarıdaki örneklerde bütün dizilerin eleman sayısı üçtür ve dördüncü elemana ulaşmak istersek programımız çalışma zamanında hata verir. Bu şekilde dizi elemanlarına değişken ve ifadeler de atanabilir.

`int[] dizi=new int[20];` satırında 20'nin illaki sabit olmasına gerek yoktur. Değişken ya da ifade de olabilir. Örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      int a=Convert.ToInt32(Console.ReadLine());
      int[] dizi=new int[a+5];
      dizi[5]=30;
      Console.Write(dizi[5]);
   }
}
```

- Eleman sayısı belirlenen bir dizinin eleman sayısı daha sonra değiştirilemez.
- Birden fazla dizi aşağıdaki gibi tanımlanabilir:

```
int[] dizi1=new int[10], dizi2=new int[20];
```

veya

```
int[] dizi1, dizi2;
```

İkincisinde tanımlanan dizinin elemanlarına henüz erişilemez.

foreach

foreach yalnızca dizilere uygulanabilen bir döngü yapısıdır. Kullanımı şu şekildedir:

```
int[] dizi={3,2,6,7};
foreach(int eleman in dizi)
   Console.WriteLine(eleman);
```

Burada `dizi` dizisinin bütün elemanları teker teker ekrana yazdırılıyor.

> foreach döngüsüyle dizi elemanlarının değerini değiştiremeyiz, sadece ekrana yazdırmak gibi "read-only" işler yapabiliriz.

Çok boyutlu diziler

Çok boyutlu diziler kısaca her bir elemanı bir dizi şeklinde olan dizilerdir, matris dizileri (düzenli diziler) ve düzensiz diziler olmak üzere ikiye ayrılır.

Matris diziler

Her bir dizi elemanının eşit sayıda dizi içerdiği dizilerdir.

İki boyutlu diziler(matrisler)

3X2 boyutunda iki boyutlu bir matris dizi aşağıdaki gibi tanımlanabilir:

```
int[,] dizi=new int[3,2];
```

veya

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

İkinci dizinin elemanları indekslerine göre aşağıdaki gibidir.
dizi[0,0] → 1
dizi[0,1] → 2
dizi[1,0] → 3
dizi[1,1] → 4
dizi[2,0] → 5
dizi[2,1] → 6
Bu diziyi matris olarak aşağıdaki gibi gösterebiliriz:

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

İkiden fazla boyutlu matris diziler

Üç boyutlu bir dizi:

```
int[,,] dizi=new int[3,2,2];
```

veya

```
int[,,] dizi={{{1,2},{3,4}},{{5,6},{7,8}},{{9,10},{11,12}}};
```

Bu dizinin indekslerine göre elemanlarıysa;
dizi[0,0,0] → 1
dizi[0,0,1] → 2
dizi[0,1,0] → 3
dizi[0,1,1] → 4
dizi[1,0,0] → 5
dizi[1,0,1] → 6
dizi[1,1,0] → 7
dizi[1,1,1] → 8
dizi[2,0,0] → 9
dizi[2,0,1] → 10
dizi[2,1,0] → 11
dizi[2,1,1] → 12

- foreach döngüsüyle iç içe döngü kurmaya gerek kalmadan her çeşit boyutlu matris dizinin elemanlarına ulaşılabilir.
- Eğer dizilerin elemanlarını değiştirmemiz gerekiyorsa iç içe for döngüsü kurmamız gerekir. Örnek:

```
int[,,] dizi={{{1,2},{3,4}},{{5,6},{7,8}},{{9,10},{11,12}}};
for(int i=0;i<3;i++)
   for(int j=0;j<2;j++)
      for(int k=0;k<2;k++)
         dizi[i,j,k]=20;
```

Bu programda dizinin bütün elemanlarının değerini 20 ile değiştirdik.

Düzensiz diziler

Her bir dizi elemanının farklı sayıda eleman içerebileceği çok boyutlu dizilerdir.

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

Birinci satırda 3 satırı olan ancak sütun sayısı belli olmayan iki boyutlu bir dizi tanımlanıyor. İkinci, üçüncü ve dördüncü satırda da bu iki boyutlu dizinin her bir satırının kaç sütun içerdiği ayrı ayrı belirtiliyor.

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

- Düzensiz dizilerin elemanlarına, örneğin 0,0 indeksine `dizi[0][0]` yazarak erişebiliriz.
- Düzensiz dizilerde foreach döngüsü sadece dizi adını yazarak çalışmaz. Ana düzensiz dizinin her bir elemanı için farklı bir foreach döngüsü başlatılmalıdır.
- Şimdiye kadar öğrendiğimiz şekilde düzensiz dizilerin elemanlarını iç içe for döngüsüyle değiştiremeyiz. Çünkü her satır farklı sayıda sütun içerebileceği için satırların sütun sayısı dinamik olarak elde edilmelidir. Bunun için C#'ın System isim alanındaki Array sınıfına ait metotları vardır ve her diziyle kullanılabilirler.

Dizilerle kullanılabilen metotlar

GetLength()

`x.GetLength(y)` şeklinde kullanılır. Herhangi bir dizinin eleman sayısını int olarak tutar. x diziyi, y o dizinin hangi boyutunun eleman sayısının hesaplanacağını belirtir. Örnekler:

```
int[] dizi={1,4,7,9};
Console.Write(dizi.GetLength(0));
```

Bu program ekrana 4 yazar.

```
int[,] dizi={{2,4,2},{7,10,4},{7,12,6},{2,1,12}};
byte a=dizi.GetLength(1);
Console.WriteLine(a);
```

Bu program ekrana 3 yazar.

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

Bu program dizinin bütün elemanlarını teker teker ekrana yazar.

CreateInstance metodu ile dizi tanımlama

Şimdiye kadar öğrendiğimiz dizi tanımlama yöntemlerinin yanında başka dizi tanımlama yöntemleri de vardır.

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

Burada int türünden 5 elemanlı `dizi` adında bir dizi tanımlandı ve dizinin her bir elemanına int türünün varsayılan değeri atandı.

```
Array dizi=Array.CreateInstance(typeof(int),3,2,5);
```

Burada 3X2X5 boyutunda int türünden 3 boyutlu bir dizi oluşturduk.

```
int[] dizi1=new int[5]{2,3,6,8,7};
Array dizi2=Array.CreateInstance(typeof(int),dizi1);
```

Burada 2X3X6X8X7 boyutunda beş boyutlu bir dizi oluşturduk.

> CreateInstance yöntemiyle oluşturulan dizilere `DiziAdi[0,4]` gibi bir yöntemle erişilemez. Şimdi bir örnek yapalım:

```
using System;
class Diziler
{
   static void Main()
   {
      Array dizi=Array.CreateInstance(typeof(int),5,4,3);
      for(int i=0;i<=dizi.GetUpperBound(0);i++)
         for(int j=0;j<=dizi.GetUpperBound(1);j++)
            for(int k=0;k<=dizi.GetUpperBound(2);k++)
               dizi.SetValue(i+j+k,i,j,k);
      for(int i=0;i<=dizi.GetUpperBound(0);i++)
         for(int j=0;j<=dizi.GetUpperBound(1);j++)
            for(int k=0;k<=dizi.GetUpperBound(2);k++)
               Console.WriteLine(dizi.GetValue(i,j,k));
   }
}
```

Daha önce CreateInstance yöntemiyle oluşturulan dizilere `DiziAdi[0,4]` gibi bir yöntemle erişilemeyeceğini söylemiştik. İşte bunun için çeşitli metotlar vardır.
GetUpperBound: Bir dizinin son indeks numarasını verir.
SetValue: Bir dizinin belirli bir indeksini belirli bir değerle değiştirir.
GetValue: Bir dizinin belirli bir indeksini tutar.

- Bu metotların kullanımları yukarıdaki örnek programda verilmiştir.
- Bu metotlar normal şekilde oluşturulan dizilerle de kullanılabilir.

Dizileri kopyalamak

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

Burada dizi1'in tüm elemanları dizi2'ye 3. indeksten itibaren kopyalanıyor.

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

Burada 3 tane eleman dizi1'den dizi2'ye kopyalanır. Kopyalama işlemi 0. indeksten başlar.

```
int[] dizi1={1,2,3,4,5,6,7};
int[] dizi2=new int[10];
Array.Copy(dizi1,2,dizi2,7,3);
```

Burada dizi1'in 2. indeksinden itibaren 3 eleman, dizi2'ye 7. indeksten itibaren kopyalanıyor.

Dizileri sıralama

Örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      Array metinsel=Array.CreateInstance(typeof(string),8);
      metinsel.SetValue("Bekir",0);
      metinsel.SetValue("Mehmet",1);
      metinsel.SetValue("Tahir",2);
      metinsel.SetValue("Yusuf",3);
      metinsel.SetValue("Yunus",4);
      metinsel.SetValue("Gökçen",5);
      metinsel.SetValue("Şüheda",6);
      metinsel.SetValue("Arzu",7);
      Console.WriteLine("Sırasız dizi:");
      foreach(string isim in metinsel)
         Console.Write(isim+" ");
      Console.WriteLine("\n\nSıralı dizi:");
      Array.Sort(metinsel);
      foreach(string isim in metinsel)
         Console.Write(isim+" ");
   }
}
```

Başka bir örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      Array sayisal=Array.CreateInstance(typeof(int),8);
      sayisal.SetValue(200,0);
      sayisal.SetValue(10,1);
      sayisal.SetValue(6,2);
      sayisal.SetValue(3,3);
      sayisal.SetValue(1,4);
      sayisal.SetValue(0,5);
      sayisal.SetValue(-5,6);
      sayisal.SetValue(12,7);
      Console.WriteLine("Sırasız dizi:");
      foreach(int sayi in sayisal)
         Console.Write(sayi+" ");
      Console.WriteLine("\n\nSıralı dizi:");
      Array.Sort(sayisal);
      foreach(int sayi in sayisal)
         Console.Write(sayi+" ");
   }
}
```

Dizilerde arama

Örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      string[] dizi={"ayşe","osman","ömer","yakup","meltem"};
      Array.Sort(dizi);
      Console.Write(Array.BinarySearch(dizi,"osman"));
   }
}
```

BinarySearch metodu, bir nesneyi bir dizi içinde arar, eğer bulursa bulduğu nesnenin indeksini tutar, bulamazsa negatif bir sayı tutar. BinarySearch'ü kullanabilmek için diziyi daha önce Sort ile sıralamalıyız. Başka bir örnek:

```
using System;
class Diziler
{
   static void Main()
   {
      string[] dizi=
{"ayşe","osman","ömer","yakup","meltem","rabia","mahmut","zafer","yılmaz","çağlayan

      Array.Sort(dizi);
      Console.Write(Array.BinarySearch(dizi,3,4,"yakup"));
   }
}
```

BinarySearch burada 3. indeksten itibaren 4 eleman içinde "yakup"u arar. Bulursa indeksini tutar. Bulamazsa negatif bir sayı tutar.
UYARI: Yalnızca tek boyutlu diziler Sort ile sıralanabilir, dolayısıyla da çok boyutlu dizilerde hem Sort ile sıralama hem de BinarySearch ile arama yapmak imkansızdır.

Diğer metotlar

```
Array.Clear(dizi,1,3);
```

Bu kod `dizi` dizisinin 1. indeksinden itibaren 3 indeksini sıfırlar (varsayılan değere döndürür).

```
Array.Reverse(dizi);
```

Bu kod `dizi` dizisinin tamamını ters çevirir.

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

Bu kod `dizi` dizisinin 1. indeksten itibaren 3 elemanını ters çevirir.