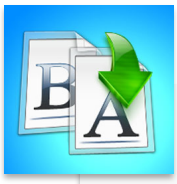
DİZİLER



İÇİNDEKİLER

- Dizi Tanımlama
- Diziye Değer Atama
- Dizi Kopyalama
- Diziye Eleman Ekleme
- Diziden Eleman Silme
- Dizilerde Sıralama
- Çok Boyutlu Diziler
- Matris Diziler
- Düzensiz Diziler

HEDEFLER

- Bu üniteyi çalıştıktan sonra;
  - Dizi yapısını anlayabilecek ve dizi tanımlayabilecek,
  - Diziler üzerinde ekleme, silme, kopyalama ve sıralama gibi işlemleri yapabilecek,
  - Çok boyutlu dizi kavramını bilecek,
  - Matris ve düzensiz dizi kavramlarını ayırt edebilecek,
  - Programlamada dizilere ait metotların kullanımı hakkında genel bilgi sahibi olabileceksiniz.

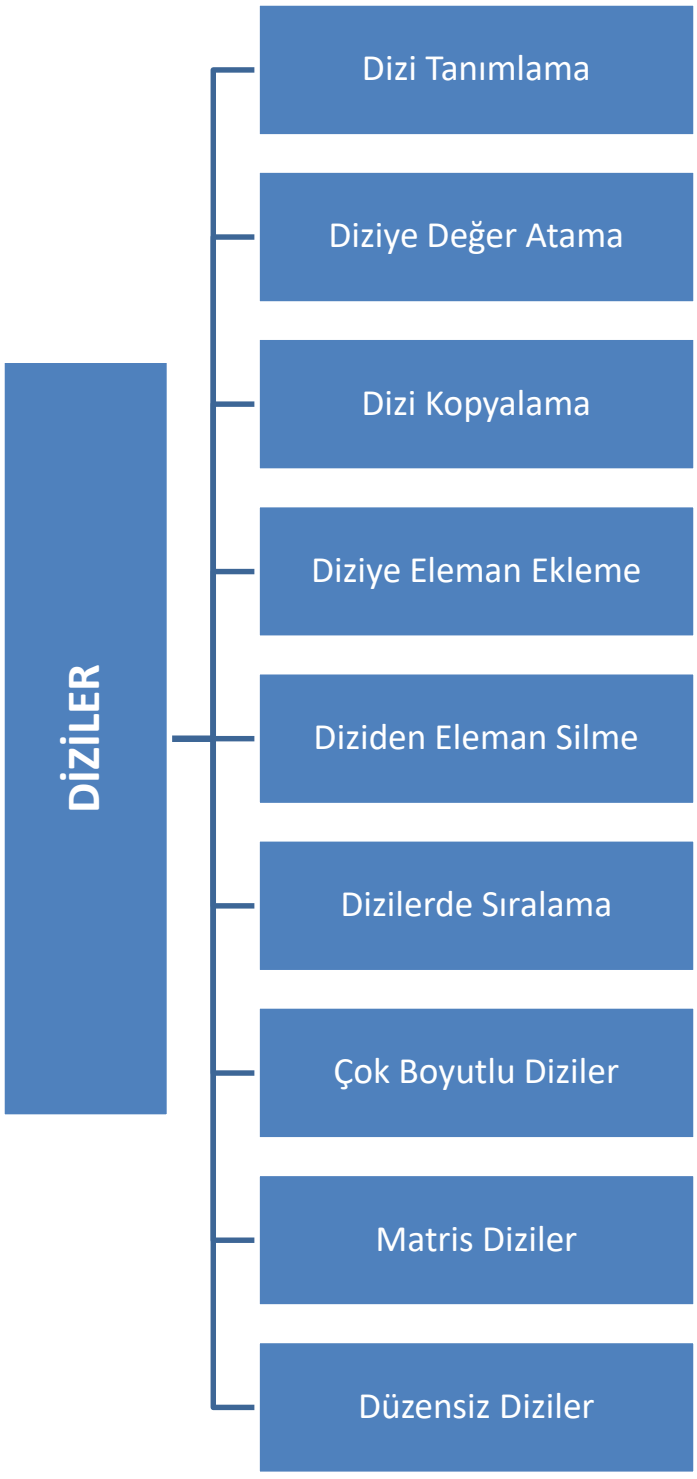
Atatürk Üniversitesi
Açıköğretim Fakültesi

NESNE TABANLI PROGRAMLAMA I

Dr. Orhan ÇELİKER

ÜNİTE

11


DİZİLER
Dizi Tanımlama
Diziye Değer Atama
Dizi Kopyalama
Diziye Eleman Ekleme
Diziden Eleman Silme
Dizilerde Sıralama
Çok Boyutlu Diziler
Matris Diziler
Düzensiz Diziler

GİRİŞ

Diziler, programlamada sıklıkla kullanılan ve ihtiyaç duyulan veri yapılarındandır. Diziler verileri gruplayarak saklamak için kullanılır. Özellikle, aynı türden verilerin sıralanarak gruplandırılmasında kullanılan dizilerin eleman sayıları sabit olarak belirlenir. Söz konusu eleman sayısının değiştirilmesi istendiğinde ise *Array.Resize* metodu kullanılır. Sevilen kitaplar veya arabalardan oluşan elemanların tek bir değişkende toplanması dizilere örnek olarak verilebilir.

Diziler, programlamada verileri kolay bir şekilde organize etme, sıralama veya düzenleme gibi çeşitli avantajlar sunar. Bunun yanı sıra dizi kullanımı, kodun okunabilirliğini artırması ve bazı karmaşık problemlerin çözümünü kolaylaştırması açısından avantaj sağlamaktadır.

Bu ünitede, C# programlama dilinde dizilerle ilgili temel işlemlerden dizi tanımlama, dizilere değişken ekleme, dizilerden eleman silme, dizileri birleştirme, dizileri sıralama, diziden kesit alma, çok boyutlu diziler ve dizi nesneleri konularına değinilecektir.

DİZİ TANIMLAMA

Bir diziye ait elemanların tamamı aynı tipte olmalıdır.

Diziler, her bir değer için ayrı değişken tanımlamak yerine birden çok değeri tek bir değişkende tutmak için kullanılır. Örneğin haftanın günlerinin bir değişkende tutulmak istendiği bir durumda her bir gün için gun1, gun2, gun3 gibi ayrı ayrı değişken tanımlanmalıdır. Ancak 7 elemanlı bir dizi kullanılarak tek bir değişkende haftanın tüm günleri saklanabilir. Diziler, C# programlama dilinde *System.Array* isim uzayı (namespace) altında yer alır. Birden çok değerin bir değişken üzerinde saklanabilmesi ve bu değerlerin sıralanarak gruplandırılabilmesi dizilerin başlıca kullanım amacıdır.

Dizi tanımlanırken birden fazla yöntem izlenebilir. Tanımlamada dizi değişkeninin veri tipi, dizi adı ve eleman sayısı belirtilir. Dizi tanımlama ve bildirimin aynı satırda yapıldığı söz dizimi için aşağıdaki gibi bir yapı kullanılabilir:

tip[ ] dizi adı = new tip[büyüklük]

Yukarıdaki yapıda yer alan tip, diziyi oluşturan elemanların değişken tipini belirlemek için kullanılır. Bu yapıda kullanılan *new* anahtar sözcüğü ise dizi değişkenine bellekte yer ayrılmasını sağlar ve köşeli parantezler içerisinde dizinin kaç elemandan oluşacağı belirtilir. Böylece dizinin büyüklüğü anlaşılır ve dizi değişkenine dinamik bir bellek alanı ayrılır. Aşağıda *ornekdizi* adında, *int* tipinde ve 7 elemandan oluşan bir dizi değişkeni tanımlama örneği verilmiştir.

```
int[] ornekdizi = new int[7];
```

Örnekten de görüleceği gibi *int* tipinde *ornekdizi* adında bir dizi bildirimi yapılmış ve bellekte *int* tipinde 5 elemanın saklanması için bir alan ayrılmıştır.

Dizi tanımlaması ve bildirimi aynı satırda yapılabildiği gibi ayrı ayrı da yapılabilir. Bunun için aşağıdaki örnek söz dizimi kullanılabilir.

```
int[] ornekdizi;
ornekdizi = new int[7];
```

Yukarıdaki örnekte ilk olarak ornekdizi adında bir değişken bildirimi yapılmıştır. Ardından bu değişken için new anahtar sözcüğü kullanılarak 7 elemanın saklanabileceği bellek alanı ayrılarak bir diziye bağlanmıştır. Dizi tanımlanırken bildirimin ve tanımlananın aynı satırda ya da farklı satırlarda yapılması tamamen yazım tercihidir, yapılmak istenen işlev açısından herhangi bir farklılık oluşturmamaktadır.

Her iki söz dizimi örneğinden de görüldüğü gibi bir dizinin kaç elemandan oluşacağı bilgisi köşeli parantez [ ] operatörü kullanılarak belirtilir. Köşeli parantezler arasına yazılan sayısal değer, dizinin kaç elemandan oluşacağını belirtmektedir. Buna göre yukarıda verilen örneklerde *ornekdizi* adında 7 elemanlı *integer* türünde bir dizi tanımlanmıştır.

Dizi tanımlanırken *new* anahtar sözcüğünün kullanılması şart değildir. Bu ifadenin kullanılması, temel veri türleri için varsayılan bir ilk değer atanmasını sağlamaktadır. Örneğin *new* anahtar sözcüğü kullanılarak *int* tipi için *0*, *bool* tipi için *false* varsayılan değerleri atanabilir. Diziler, *new* ifadesi kullanılmadan aşağıdaki örnek söz dizimi ile de tanımlanabilir:

```
int[] sayilar = {2, 4, 6, 8};
```

Dizi elemanları soldan sağa doğru indekslenir ve her bir diziye bir indeks numarası atanır. Yukarıda bulunan örnekteki tanımlamada diziye otomatik olarak bellek alanı ayrılmış ve *new* anahtar sözcüğünün kullanılmasına gerek kalmamıştır. Dolayısıyla önceden elemanları belli olan değişkenler için dizinin bu şekilde tanımlanması daha avantajlıdır.

DİZİYE DEĞER ATAMA

Bir dizi tanımlandıktan sonra öncelikle dizi değişkenine değer atanmalıdır. Dizi değişkenine değer atamak için dizi elemanının indeks değeri köşeli parantez içine yazılır ve istenilen değer atanır.

Aşağıdaki örnekte *string* tipinde 7 elemanlı bir dizi değişkeni tanımlanmıştır. Ardından *[ ]* operatörü kullanılarak dizi elemanlarına erişilmiş ve elemanlara değer ataması yapılmıştır. Dizi değişkeninin elemanlarına değer ataması yapılırken dizi indeks numarası her zaman 0'dan başlar ve 1, 2, … şeklinde devam eder. Aşağıdaki örnekte de gunler dizisinin 1. elemanına 0 indeks numarası ile erişilmiş ve "Pazartesi" değeri atanmıştır. Sonrasında 1'den 6'ya kadar indeks numaraları kullanılarak dizinin tüm elemanlarına değer ataması yapılmıştır.

```
string[] gunler = new string[7];
gunler[0] = "Pazartesi";
gunler[1] = "Salı";
gunler[2] = "Çarşamba";
gunler[3] = "Perşembe";
```

```
gunler[4] = "Cuma";
gunler[5] = "Cumartesi";
gunler[6] = "Pazar";
```

Yukarıdaki örnekte dizi elemanları belli olduğundan diziye değer ataması aşağıdaki gibi dizi tanımlanırken de yapılabilir. Aşağıda günler adında 7 elemanlı bir dizi değişkeni tanımlanmış ve aynı bildirimde dizi elemanlarına değer ataması yapılmıştır.

Dizi değişkeninin elemanlarına değer ataması yapılırken en çok dikkat edilmesi gereken nokta, değişken tipine göre değer atamaktır.

```
string[] gunler = {"Pazartesi", "Salı", "Çarşamba",
"Perşembe", "Cuma", "Cumartesi", "Pazar"};
```

Dizi değişkeninin elemanlarına değer ataması yapılırken en çok dikkat edilmesi gereken nokta, değişken tipine göre değer atamaktır. Bunun sebebi her tip kendi tipinden olan değerleri kabul etmektedir. Örneğin string tipinde tanımlanan bir dizi değişkenine int tipinde bir değer atanamaz veya decimal tipindeki bir dizi değişkenine bool tipinde bir değer atanamaz. Dizi kullanımında dikkat edilmesi gereken başka bir nokta da dizi elemanlarına erişilirken kesinlikle tam sayı kullanılmalıdır. Bu hususlar oldukça önemlidir uyulmadığı durumlarda ise kod hatasıyla karşılaşılır.

DİZİ KOPYALAMA

Bazı durumlarda bir dizinin içeriği başka bir diziye kopyalanabilir veya iki dizinin içerikleri birleştirilebilir. Bu tür durumlar için döngülerin yer aldığı algoritmalar kullanılabilir. Ancak yazılım geliştirme ortamlarının sunduğu avantaşlar sayesinde dizilerden kesit alma, sıralama, dizi birleştirme gibi karmaşık işlemler programlama dilinde hazır bulunan metot ve fonksiyonlar aracılığıyla kolaylıkla gerçekleştirilebilir.

C# programlama dilinde Copy(), CopyTo() ve Clone() metodları yardımıyla iki dizinin içerikleri birbirine kopyalanabilir veya diziler tek bir dizide birleştirilebilir.

C# programlama dilinde *Copy(), CopyTo()* ve *Clone()* metodları yardımıyla iki dizinin içerikleri birbirine kopyalanabilir veya diziler tek bir dizide birleştirilebilir. *CopyTo()* metodu kullanılarak bir dizinin içeriğini başka bir diziye kopyalamak için kopyası alınacak kaynak dizi, indeks numarası ve hedef dizi belirtilir. Bir diziden başka bir diziye içerik kopyalama işleminin kod örneği aşağıda verilmiştir.

```
int[] kaynakdizi = { 1, 3, 5 };
int[] hedefdizi = new int[3];
kaynakdizi.CopyTo(hedefdizi, 0);
```

Örnekten de görüldüğü gibi *CopyTo()* metodu iki parametre almaktadır. ilk parametre olarak kopyalama yapılacak hedef dizinin adını, ikinci parametre olarak ise kopyalama işleminin başlayacağı indeks numarasını almaktadır. Bu örneğe göre *kaynakdizi* adındaki dizi değişkeni *hedefdizi* adındaki 3 elemanlı int tipindeki dizi değişkenine 0. indeksten başlanarak kopyalanmıştır.

Dizi içeriğinin kopyalanması için kullanılan bir başka yöntem de *Copy()* metodudur. Bu metot *Array* sınıfı altında yer alır ve kullanımında 3 parametre tanımlanır. İlk parametrede kaynak dizi adı, ikinci parametrede hedef dizi adı ve

üçüncü parametrede ise kopyalanacak eleman sayısı bildirilir. *Copy()* metodunun kullanımına ait kod örneği aşağıdaki gibidir:

```
int[] kaynakdizi = { 1, 3, 5, 7, 9, 11 };
int[] hedefdizi = new int[6];
Array.Copy(kaynakdizi, hedefdizi, 6);
```

Örnekten de görüldüğü gibi kaynakdizi adındaki dizinin 6 elemanı bulunmaktadır. Bu dizinin 6 elemanı hedefdizi adındaki diziye kopyalanmak istenmiş ve bunun için Array sınıfına ait Copy() metodu kullanılmıştır. Copy() metodunun parametrelerinde ise parantez içinde hangi diziden (kaynakdizi), hangi diziye (hedefdizi) kaç elemanın kopyalanacağı bilgisi bildirilmiştir. Bu kod örneği çalıştırıldığında kaynakdizi adındaki diziye ait 1, 3 ,5, 7, 9 ve 11 elemanları hedefdizi adındaki dizi değişkenine kopyalanacaktır.

Dizi içeriğini kopyalamanın bir başka yolu da Clone() metodunu kullanmaktır. Bu metot *System.Array* sınıfı altında yer alır ve bir dizinin kopyasını oluşturmak için kullanılır. Ancak *Clone()* metodu *object* tipinde bir çıktı döndürür ve kod hatası yaşanmaması için bu metot kullanıldığında tip dönüşümü yapılmalıdır. *Clone()* metodu kullanılarak dizi kopyalama işleminin yapıldığı kod örneği aşağıda verilmiştir.

```
int[] kopyalanacakdizi = { 1, 3, 5, 7, 9, 11 };
int[] hedefdizi = (int[])kopyalanacakdizi.Clone();
```

Sonuç olarak bir dizinin içeriğinin kopyalanması ya da başka bir diziyle birleştirilmesi gerektiğinde döngülerin kullanıldığı çeşitli algoritmaları yazmak yerine yazılım geliştirme ortamlarının sunduğu hazır metotlar kullanılabilir.

Dizilere değer atama işlemi dizi tanımlanırken yapılabildiği gibi dizi tanımlandıktan sonra SetValue metodu kullanılarak da yapılabilir.

DİZİYE ELEMAN EKLEME

Dizilere değer atama işlemi dizi tanımlanırken yapılabildiği gibi dizi tanımlandıktan sonra *SetValue* metodu kullanılarak da yapılabilir. Bazı durumlarda dizi tanımlandıktan sonra diziye yeni eleman ekleme ihtiyacı doğabilir. Bu tür durumlarda C# programlama dilinde *Array* sınıfının altında yer alan *SetValue* metodu kullanılır. Bu metot kullanılırken öncelikle dizi değişkeni adı yazılır ve parametre olarak *eklenecek elemanın değeriyle indeks numarası* bildirilir. *SetValue* metodunun kullanımına ilişkin örnek kod aşağıdaki gibidir:

```
string[] ogrenciler = new string[4];
ogrenciler.SetValue("Yiğit", 0);
ogrenciler.SetValue("Zafer", 1);
ogrenciler.SetValue("Çınar", 2);
ogrenciler.SetValue("Fatih", 3);
```

Dizilere değer eklemenin bir başka yöntemi de = operatörünün kullanılmasıdır. Bu yöntemde dizi elemanının indeks numarası belirtilir ve =

operatörü kullanılarak değer ataması yapılır. *SetValue* metodu kullanılmadan diziye elaman eklemek için kullanılan kod örneği aşağıdaki gibidir:

```
string[] ogrenciler = new string[4];
ogrenciler[0] = "Yiğit";
ogrenciler[1] = "Zafer";
ogrenciler[2] = "Çınar";
ogrenciler[3] = "Fatih";
```

Bu iki yöntemle de dizi değişkenlerine eleman eklenebilir. Bu yöntemler arasında ciddi bir düzeyde performans ve kullanım farklılığı bulunmadığından hangisinin tercih edileceği geliştiricinin kendisine bağlıdır.

DİZİDEN ELEMAN SİLME

Dizilere yeni eleman eklemek gibi dizilerden mevcut eleman ya da elemanların silinmesi de programlamada sıkça karşılaşılan durumlardandır. Bu işlem için C# programlamada *Array* sınıfının altında yer alan *Clear()* metodu kullanılır. Bu metot sözdiziminde 3 parametre alır. Bunlardan ilki elemanları sıfırlanacak dizi değişkenini, ikincisi içerik silme işleminin başlayacağı elemanı ve üçüncüsü de içeriği silinecek eleman sayısını ifade eder. Bu metot kullanılırken aşağıdaki sözdizimine uyulması gerekmektedir:

Diziden eleman silmek için C# programlamada Array sınıfının altında yer alan Clear() metodu kullanılır.

Array.Clear(Dizi Adı, Başlangıç İndeks Numarası, Silinecek Eleman Sayısı)

Clear() metoduna ait bu söz dizimi kullanılarak aşağıdaki gibi bir örnek verilebilir:

```
string[] dersler = new string[4];
dersler[0] = "Görsel Programlama";
dersler[1] = "Web Tasarımı";
dersler[2] = "Nesne Yönelimli Programlama";
dersler[3] = "İnternet Programcılığı";
Array.Clear(dersler, 0, 2);
```

Örnekten de görüldüğü üzere *Clear()* metodunun ilk parametresinde dizi değişkeninin adı (dersler) kullanılmıştır. Bu, gerçekleştirilecek silme işleminin dersler dizisi üzerinde yapılacağı anlamına gelmektedir. İkinci parametrede yer alan 0 ifadesi, silme işleminin dizinin ilk elemanından başlayacağını üçüncü parametredeki 2 ifadesi ise iki elemanın silineceğini bildirir. Buna göre yukarıdaki kod örneği çalıştırıldığında dersler dizi değişkeninin 1 ve 2. elemanlarına ait "Görsel Programlama", "Web Tasarımı" değerleri silinecektir. Bu örnekte üçüncü parametre 2 yerine 4 şeklinde bildirilirse dizinin tüm elemanlarına ait değerler silinir.

Diziden eleman silmeyle ilgili konunun daha iyi anlaşılması açısından bir de int tipinde dizi değişkeni için örnek verilmiştir. Bu örnekte de sayilar isimli dizinin ilk elemanından başlanarak 4 elemanına ait değerler silinmiştir.

```
int[] sayilar = new int[5];
sayilar[0] = 5;
sayilar[1] = 10;
sayilar[2] = 15;
sayilar[3] = 20;
sayilar[4] = 25;
Array.Clear(sayilar, 0, 3);
```

Örnekteki kod örneği çalıştırıldığında öncelikle 5 elemandan oluşan sayilar isimli bir dizi oluşturulur ve dizi elemanlarına değerler atanır. Ancak son satırda yer alan Clear() metodu ile dizi değişkeninin ilk üç elemanına ait değerler silinmiş, dizinin 4. ve 5. elemanlarına ait değerleri kalmıştır.

Diziden eleman silme işlemiyle dizi için bellekte ayrılan alan silinmez sadece elemanların değerleri temizlenir. Silme işlemi sonrasında bellekte ayrılan alan string tipinde ise " " (boşluk), integer tipinde ise 0 (sıfır) değerini alır.

DİZİLERDE SIRALAMA

C# programlama dilinde sıralama işlemi için Array sınıfının altında Sort() metodu kullanılabilir.

Geliştirilen yazılımlarda ihtiyaca göre dizi içeriğinin sıralanması gerekebilir. Dizilerin kullanımında sıralama işlemi oldukça önemlidir. Bu işlem için geliştiriciler, çeşitli fonksiyonlar veya metotlar kullanabilirler. Ancak C# programlama dili sıralama işlemi için *Array* sınıfının altında *Sort()* metodunu sunmuştur. Bu metot tanımlanırken aşağıdaki sözdizimi kullanılır.

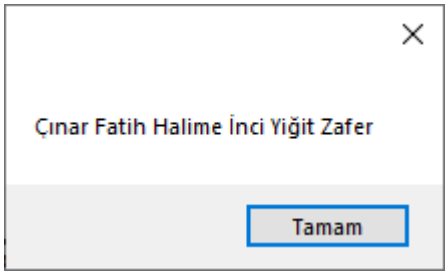
Array.Sort(Dizi Adı)

Sort() metodu sıralama yapılmak istenen diziyi parametre olarak alır ve dizi elemanlarını A'dan Z'ye ya da küçükten büyüğe sıralar. *Sort()* metodunun kullanımına yönelik oluşturulan kod örneği aşağıdaki gibidir:

```
string[] isimlistesi = new string[6];
isimlistesi[0] = "Yiğit";
isimlistesi[1] = "Zafer";
isimlistesi[2] = "Çınar";
isimlistesi[3] = "Fatih";
isimlistesi[4] = "İnci";
isimlistesi[5] = "Halime";
Array.Sort(isimlistesi);
string siralama = " ";
for (int i = 0; i < 6; i++)
{
      siralama += isimlistesi[i].ToString() + " ";
}
```

```
MessageBox.Show(siralama.ToString());
```

Örnekte 6 elemanlı isimlistesi dizisi tanımlanmış ardından dizi elemanları Sort() metodu yardımıyla alfabetik olarak sıralanmıştır. Sıralama işlemine alınan dizi elemanları bir döngüye tabi tutularak bir değişkene atanmış ve *MessageBox* ile görüntülenmiştir. Yukarıdaki örnek kod çalıştırıldığında sıralanan dizi elemanlarının görünümü Resim 12.1'deki gibidir.



Şekil 11.1. MessageBox Görünümü

Yukarıdaki örnekte verilen dizi metinsel ifadelerden oluşan *string* tipinde bir dizidir. Bunun yerine sayısal karakterlerden oluşan *integer* tipinde bir dizi oluşturulduğunda sıralama küçükten büyüğe doğru yapılırdı. Bunun nedeni *Sort()* metodu sıralanması istenen dizi değişkeninin elemanlarını varsayılan olarak küçükten büyüğe sıralar. Sıralamayı tersine çevirmek yani büyükten küçüğe sırlama işlemi yapmak için yine *Array* sınıfının altında yer alan *Reverse()* metodu kullanılmalıdır. *Reverse()* metodu *Sort()* metoduyla aynı mantıkta çalışır ve parametre olarak sıralama yapılacak dizinin adını alır. Ancak bu metot sıralama işleminin yerine sadece yapılan sıralama işleminin terse çevirerek görüntüler. *Bir başka ifadeyle Reverse() metodu kullanılarak diziye ait tüm elemanların ya da belli bir aralığın sırasını terse çevirerek görüntülemeyi sağlar.*

ÇOK BOYUTLU DİZİLER

Çok boyutlu dizi tanımlanırken virgül (,) operatörü kullanılarak dizinin boyutu belirtilir.

Diziler tek boyutlu ve çok boyutlu olmak üzere iki gruba ayrılabilir. Tek boyutlu diziler tanımlanırken sadece eleman sayısı belirtilir. Bu ünitede şimdiye kadar ele alınan başlıklar tek boyutlu dizilere örnek verilebilir. Çok boyutlu diziler ise tek boyutlu dizilere benzer biçimde tanımlanır ve kullanılır. Ancak çok boyutlu dizi tanımlanırken *virgül (,)* operatörü kullanılarak dizinin boyutu belirtilir.

Çok boyutlu dizilerde 2 boyutlu diziler sıklıkla kullanılmaktadır. Çok boyutlu dizinin elemanlarına erişmek için ise iki veya daha fazla indeks kullanılmalıdır. Çok boyutlu diziler matris ve düzensiz diziler olmak üzere iki kategoride incelenebilir.

MATRİS DİZİLER

Dikdörtgensel diziler olarak da adlandırılan matris dizilerin her satırında eşit sayıda eleman bulunur. Aşağıda iki boyutlu bir matris dizinin sözdizimi gösterilmiştir.

Değişken Tipi[,] Dizi Adı = {{x,y},{z,t}}

Bu sözdizimine göre matris dizilere aşağıdaki gibi bir örnek verilebilir.

```
string[,] ornekmatrisdizi = { { "Ocak", "Şubat" },
{ "Mart", "Nisan" }, { "Mayıs", "Haziran" } };
```

Yukarıdaki kod örneğinde 2 boyutlu 3 elemanlı dizi tanımlanmıştır. Bu yapı bir tablo şeklinde düşünülecek olursa dizinin 3 satır ve 2 sütuna sahip olduğu söylenebilir. Örnekteki diziye ait elemanların hangi indeks numarasına sahip olduklarını daha iyi anlamak için Tablo 12.1'de görselleştirme yapılmıştır.

Tablo 11.1. Matris Dizi Görünümü

ornekmatrisdizi [0] [0]	ornekmatrisdizi [0] [1]
ornekmatrisdizi [1] [0]	ornekmatrisdizi [1] [1]
ornekmatrisdizi [2] [0]	ornekmatrisdizi [2] [1]

Tablodan da görüldüğü gibi indeks numaraları *köşeli parantezler [ ] [ ]* içerisinde verilmiştir. Bu indeks numaralarından ilki hangi satıra, ikincisi ise hangi sütuna erişileceğini bildirir. Bu yapıya göre hangi elemanın hangi indeks numarasına sahip olduğu Tablo 12.2'de verilmiştir.

Tablo 11.2. Matris Dizi Elemanları İndeks Numaraları

Dizi İndeks Numarası	İlgili Eleman
ornekmatrisdizi [0] [0]	Ocak
ornekmatrisdizi [0] [1]	Mart
ornekmatrisdizi [1] [0]	Mayıs
ornekmatrisdizi [1] [1]	Şubat
ornekmatrisdizi [2] [0]	Nisan
ornekmatrisdizi [2] [1]	Haziran

Tablo incelendiğinde tıpkı tek boyutlu dizilerde olduğu gibi çok boyutlu dizilerde de her boyutun ilk elemanına 0 indeks numarasıyla erişilir. Yukarıdaki örneğe göre ornekmatrisdizi isimli dizinin ilk elemanına [0][0] indeks numarasıyla erişilir. Bu da tablonun birinci satır ve sütunundaki "Ocak" değerine denk gelmektedir. Aynı şekilde dizinin ikici satır ve ikinci sütunundaki elemanına erişmek için [1][1] indeks numarası kullanılmalıdır.

Bireysel Etkinlik

- Kayıtlı olduğunuz bölümün derslerini döneme göre gösteren çok boyutlu bir dizi oluşturunuz. Ardından bir döngü kullanarak bu dizideki elemanları farklı listbox ögelerinde görüntülenmesini sağlayınız.

DÜZENSİZ DİZİLER

Her bir satırı farklı uzunlukta olan çok boyutlu dizilere düzensiz diziler denmektedir. İç içe diziler olarak da adlandırılan düzensiz dizilerin her bir elemanının içerisinde başka bir dizinin elemanları tanımlanabilir.

Düzensiz diziler tanımlanırken her bir boyut için bir çift köşeli parantez [ ] kullanılır.

Düzensiz diziler tanımlanırken her bir boyut için *bir çift köşeli parantez [ ]* kullanılır. Örneğin iki boyutlu bir düzensiz dizi oluşturmak için kullanılan sözdizimi aşağıdaki gibidir.

DeğişkenTipi [ ][ ] Dizi Adı = new DeğişkenTipi [Büyüklük][ ]

Düzensiz diziler tanımlanırken bildirilen büyüklük parametresi dizinin kaç satırdan oluşacağını belirtir. Bu büyüklüğe göre bellekte satırlar için alan ayrılır ve böylece her bir satırın faklı uzunlukta olması sağlanır. Düzensiz bir dizinin nasıl tanımlandığı ve her bir satırının farklı uzunlukta olduğunu gösteren kod örneği aşağıda verilmiştir.

```
int[][] ornekduzensizdizi = new int[4][];
ornekduzensizdizi[0] = new int[3];
ornekduzensizdizi[1] = new int[5];
ornekduzensizdizi[2] = new int[2];
ornekduzensizdizi[3] = new int[4];
```

Yukarıdaki kod örneğinin ilk satırında ornekduzensizdizi isimli tek boyutlu ve 4 satırdan oluşan bir dizi tanımlanmıştır. Tanımlanan bu diziye ait her eleman da sonraki 4 satırda farklı birer dizi olarak tanımlanmıştır. Aşağıdaki şekilde bu örnek dizi görsel olarak temsil edilmiştir.

Şekil 11.2. Düzensiz Dizi Elemanları İndeks Numaraları

Örnekten de görüldüğü gibi dizi elemanlarına köşeli parantezlerin içindeki indeks numaraları kullanılarak erişim sağlanır ve değer ataması yapılabilir. Bu işlemde ilk köşeli parantez dizi elemanının bulunduğu satır sayısını, ikinci köşeli parantez ise erişilmesi istenen elemanın indeks numarasını içerir.

Özet

- Diziler, programlamada verileri kolay bir şekilde organize etme, sıralama veya düzenleme gibi çeşitli avantajlar sunar. Bunun yanı sıra dizi kullanımı, kodun okunabilirliğini artırması ve bazı karmaşık problemlerin çözümünü kolaylaştırması açısından avantaj sağlamaktadır.
- Diziler, her bir değer için ayrı değişken tanımlamak yerine birden çok değeri tek bir değişkende tutmak için kullanılır.
- Dizi tanımlanırken birden fazla yöntem izlenebilir. Tanımlamada dizi değişkeninin veri tipi, dizi adı ve eleman sayısı belirtilir.
- Bir dizi tanımlandıktan sonra öncelikle dizi değişkenine değer atanmalıdır. Dizi değişkenine değer atamak için dizi elemanının indeks değeri köşeli parantez içine yazılır ve istenilen değer atanır.
- *Dizi değişkeninin elemanlarına değer ataması yapılırken en çok dikkat edilmesi gereken nokta, değişken tipine göre değer atamaktır. Bunun sebebi her tip kendi tipinden olan değerleri kabul etmektedir.* Örneğin string tipinde tanımlanan bir dizi değişkenine int tipinde bir değer atanamaz veya decimal tipindeki bir dizi değişkenine bool tipinde bir değer atanamaz. Dizi kullanımında dikkat edilmesi gereken başka bir nokta da dizi elemanlarına erişilirken kesinlikle tam sayı kullanılmalıdır. Bu hususlar oldukça önemlidir uyulmadığı durumlarda ise kod hatasıyla karşılaşılır.
- Bazı durumlarda bir dizinin içeriği başka bir diziye kopyalanabilir veya iki dizinin içerikleri birleştirilebilir. Bu tür durumlar için döngülerin yer aldığı algoritmalar kullanılabilir. Ancak yazılım geliştirme ortamlarının sunduğu avantaşlar sayesinde dizilerden kesit alma, sıralama, dizi birleştirme gibi karmaşık işlemler programlama dilinde hazır bulunan metot ve fonksiyonlar aracılığıyla kolaylıkla gerçekleştirilebilir.
- C# programlama dilinde *Copy(), CopyTo()* ve *Clone()* metodları yardımıyla iki dizinin içerikleri birbirine kopyalanabilir veya diziler tek bir dizide birleştirilebilir. *CopyTo()* metodu kullanılarak bir dizinin içeriğini başka bir diziye kopyalamak için kopyası alınacak kaynak dizi, indeks numarası ve hedef dizi belirtilir.
- Dizilere değer atama işlemi dizi tanımlanırken yapılabildiği gibi dizi tanımlandıktan sonra *SetValue* metodu kullanılarak da yapılabilir. Bazı durumlarda dizi tanımlandıktan sonra diziye yeni eleman ekleme ihtiyacı doğabilir. Bu tür durumlarda C# programlama dilinde *Array* sınıfının altında yer alan *SetValue* metodu kullanılır. Bu metot kullanılırken öncelikle dizi değişkeni adı yazılır ve parametre olarak *eklenecek elemanın değeriyle indeks numarası* bildirilir.
- Dizilere yeni eleman eklemek gibi dizilerden mevcut eleman ya da elemanların silinmesi de programlamada sıkça karşılaşılan durumlardandır. Bu işlem için C# programlamada *Array* sınıfının altında yer alan *Clear()* metodu kullanılır. Bu metot sözdiziminde 3 parametre alır. Bunlardan ilki elemanları sıfırlanacak dizi değişkenini, ikincisi içerik silme işleminin başlayacağı elemanı ve üçüncüsü de içeriği silinecek eleman sayısını ifade eder.
- Geliştirilen yazılımlarda ihtiyaca göre dizi içeriğinin sıralanması gerekebilir. Dizilerin kullanımında sıralama işlemi oldukça önemlidir. Bu işlem için geliştiriciler, çeşitli fonksiyonlar veya metotlar kullanabilirler. Ancak C# programlama dili sıralama işlemi için *Array* sınıfının altında *Sort()* metodunu sunmuştur.
- Diziler tek boyutlu ve çok boyutlu olmak üzere iki gruba ayrılabilir. Tek boyutlu diziler tanımlanırken sadece eleman sayısı belirtilir. Bu ünitede şimdiye kadar ele alınan başlıklar tek boyutlu dizilere örnek verilebilir. Çok boyutlu diziler ise tek boyutlu dizilere benzer biçimde tanımlanır ve kullanılır. Ancak çok boyutlu dizi tanımlanırken *virgül (,)* operatörü kullanılarak dizinin boyutu belirtilir.
- Dikdörtgensel diziler olarak da adlandırılan matris dizilerin her satırında eşit sayıda eleman bulunur.
- Her bir satırı farklı uzunlukta olan çok boyutlu dizilere düzensiz diziler denmektedir.

DEĞERLENDİRME SORULARI

1. C# dilinde diziler hangi isim uzayı altında yer alır?
   a) System.Array
   b) System.Data
   c) System.IO
   d) System.Text
   e) System.Drawing

2. Aşağıdakilerden hangisi dizi tanımlanırken kullanılabilen deyimlerden biridir?
   a) break
   b) new
   c) create
   d) type
   e) throw

3. Aşağıdakilerden hangisi dizilerle ilgili yanlış bir ifadedir?
   a) Matris dizi çok boyutlu bir dizidir.
   b) C#'ta Copy, CopyTo ve Clone metodları ile dizi kopyalama yapılabilir.
   c) Dizi tanımlanırken new anahtar sözcüğü kesinlikle kullanılmalıdır.
   d) String tipinde bir dizi elemanı silinirse varsayılan değeri ""(boşluk) olur.
   e) Dizilerin elemanlarına tek bir isim ve indeks numarası ile ulaşılabilir.

4. C#'ta string tipinde bir dizi değişkeninin elemanlarına değer atamak için kullanılan operatör aşağıdakilerden hangisidir?
   a) []
   b) =
   c) {}
   d) ""
   e) ()

5. C#'ta sıralanmış bir dizinin tersten görüntülenmesini sağlamak için kullanılan metot aşağıdakilerden hangisidir?
   a) Insert()
   b) Reverse()
   c) Clone()
   d) Sort()
   e) CopyTo()

6. C#'ta bir diziden eleman silmek için kullanılan metot aşağıdakilerden hangisidir?
   a) Insert()
   b) CopyTo()
   c) Clear()
   d) Sort()
   e) Reverse()

7. C#'ta bir dizinin içeriğini kopyalamak için kullanılan metot aşağıdakilerden hangisidir?
   a) Insert()
   b) Clone()
   c) Clear()
   d) Sort()
   e) Reverse()

8. C#'ta çok boyutlu dizi tanımlanırken aşağıdaki operatörlerden hangisi kullanılmalıdır?
   a) ,
   b) .
   c) ;
   d) :
   e) *

9. C#'ta bir dizinin elemanına değer atamak için aşağıdaki metotlardan hangisi kullanılır?
   a) GetValue()
   b) Copy()
   c) CopyTo()
   d) Clear()
   e) SetValue()

10. Array.Sort(diziadi) kodu ile yapılmak istenen işlem aşağıdakilerden hangisidir?
    a) Dizi kopyalama
    b) Diziye eleman ekleme
    c) Diziye değer atama
    d) Dizi sıralama
    e) Dizi tanımlama

Cevap Anahtarı
1.a, 2.b, 3.c, 4.d, 5.b, 6.c, 7.b, 8.a, 9.e, 10.d

YARARLANILAN KAYNAKLAR

Aktaş, Volkan, (2021), Her Yönüyle C# 9.0, KODLAB, İstanbul.

Clark, Dan, (2013), Beginning C# Object-Oriented Programming 2nd Edition, Apress, New York.

Grıffıths, Ian, (2013), Programming C# 5.0, O'Reilly Media, California. KIZILÖREN, Tevfik, (2013), Her Yönüyle C, KODLAB, İstanbul.

https://www.w3schools.com/cs/cs_arrays.php. [Erişim Tarihi: 05.09.2021]

Schıldt, Herbert, (2005), The Complete Reference C#, çev. Duygu Arbatlı Yağcı, Alfa Basım Yayım Dağıtım, İstanbul.

Sharp, John, (2009), Microsoft Visual Studio 2008 Step By Step, çev. Ümit Tezcan, Arkadaş Yayınevi, Ankara.

Skeet, Jon, (2014), C# in Depth 3rd Edition, Manning Publication Co, New York.