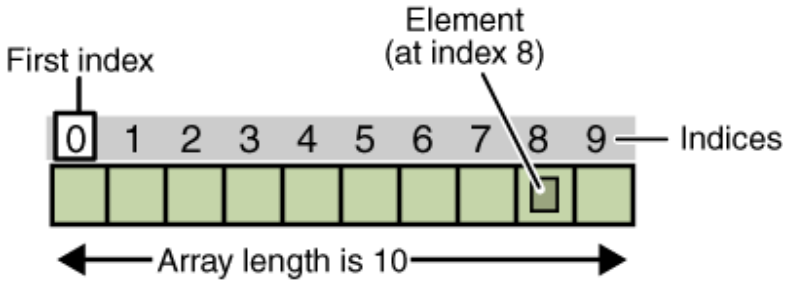
Diziler

Diziler programlama dilleri için önemli kavramlardır. Eğer diziler olmasaydı uygulama içerisinde birden fazla değişkene sahip olmak istediğimizde her seferinde bunlar için bir tanımlama yapmak zorunda kalırdık.

Dizi kavramı tanım olarak aynı tipte nesnelerin sıralı bir şekilde bellekte saklanmasını ifade eder. Burada onemli nokta nesnelerin aynı tipte olmasıdır. Örneğin int tipinde oluşturulan bir dizi içerisinde sadece int tipinde değerler taşıyabilir. Bunların tanımları bellekte sıralı olarak bir indeks numarası ile saklanır.

Aslında farketmesekte uygulama içerisinde collection framework gibi bir çok yapı diziler ile tasarlanmıştır. Yani List kullanan her şey alt tarafta diziyi de kullanır.

Dikkat : C/C++ türevi dillerin hemen hemen hepsinde dizi indeksleri 0' dan başlamaktadır. Bu durumda 4 elemanlı bir dizinin elemanlar 0 – 1 – 2 – 3 şeklindedir.



Peki Java tarafında dizilerin kullanımı nasıldır ? Diziler öncelikle tanımlanmalı ve uzunlukları ile birlikte yaratılmalıdırlar. Eğer yaratılan dizi ilkel tipteyse bellek üzerinde ilk değerlerini alırlar. Refernans tipindeyse atamaları beklenir. Aşağıda örnek dizi tanımları bulunmaktadır.

```
int[] diziBir = new int[6];
double[] diziIki = new double[10];
String[] diziUc = new String[2];
Ogrenci[] diziDort = new Ogrenci[8];
```

Burada 4 farklı tipte dizi yarattık ilk iki dizi ilkel tipte, diğer ikisi ise nesne tipindeler.

Not : Java içerisindeki dizi tipleri sadece Java içerisinde geçen sınıflarda olmak zorunda değildir. Örnek olduğu gibi Ogrenci veya Ogretmen gibi kendi sınıf tiplerimizde dizilerde olabilir.

Dizi değerlerinin atanması

Dizi değerlerini atamak için iki yöntem vardır birincisi direkt ilgili indeks üzerinden ilgini alana atama yapmak diğeri ise diziyi tanımlarken değerlerini vermek.

Birinci yöntemi aşağıdaki gibi örneklendirebiliriz.

```
public class Ornek {
	public static void main(String[] args) {
		String[] sehirler = new String[4];
		sehirler[0] = "Ankara";
		sehirler[1] = "Istanbul";
		sehirler[2] = "İzmir";
		sehirler[3] = "Balıkesir";
	}
}
```

Buna göre 4 elemanlı bir dizinin 0, 1, 2 ve 3. Elemanlarına atama yapmış oluyoruz. Eğer aşağıdaki gibi bir atama yapıyor olsaydık bizim için hatalı bir durum oluşacakı.

```
sehirler[4] = "Adana";
```

Bu hatanın nedeni 4 uzunluğunda bir dizinin son elemanının 3 olmasıdır. 4. elemana erişmeye çalıştığınız zaman üretilen hata aşağıdaki gibidir.

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
	at Ornek.main(Ornek.java:7)
```

Not : Hata tiplerini ve bunları yönetmeyi ileriki bölümlerde inceliyor olacağız.

Dizilere atama yapmanın diğer yolu aşağıdaki gibi diziyi oluşturken değerlerini vermektir.

```
public class Ornek {
	public static void main(String[] args) {
		String[] sehirler = {"Ankara", "İstanbul", "İzmir", "Balıkesir"};
	}
}
```

Buna göre dizinin uzunluğu verilmez. Uzunluk olarak atanan değerlerin sayısını alır. Buna göre yukarıdaki dizinin uzunluğu 4' tür.

Dizilere ulaşım

Atanan dizi değerlerine ulaşım için elemanın indeks numarasına ihtiyacımız vardır. Aşağıdaki örnekte atama yapılmış dizinin elemanları bulunmaktadır.

```
public class Ornek {
	public static void main(String[] args) {
		String[] sehirler = new String[4];
		sehirler[0] = "Ankara";
		sehirler[1] = "Istanbul";
		sehirler[2] = "İzmir";
		sehirler[3] = "Balıkesir";

		System.out.println("1. Şehir : " + sehirler[0]);
```

```
            System.out.println("2. Şehir : " + sehirler[1]);
            System.out.println("3. Şehir : " + sehirler[2]);
            System.out.println("4. Şehir : " + sehirler[3]);
      }
}
```

Bu kodun çıktısı aşağıdaki gibi olacaktır.

```
1. Şehir : Ankara
2. Şehir : Istanbul
3. Şehir : İzmir
4. Şehir : Balıkesir
```

Peki dizimizin uzunluğu 100 ise içerisindeki elemanları tek tek yazarak mı ulaşacağız ? Bunun yerine dizinin uzunluğunu alarak döngü ile yazdırma işlemi yapabiliriz. Dizi uzunluğunu almak için aşağıdaki kod örneğindeki gibi length özelliği kullanılabilir.

```
public class Ornek {
      public static void main(String[] args) {
            String[] sehirler = new String[4];
            sehirler[0] = "Ankara";
            sehirler[1] = "Istanbul";
            sehirler[2] = "İzmir";
            sehirler[3] = "Balıkesir";

            for (int i = 0; i < sehirler.length; i++) {
                  System.out.println(sehirler[i]);
            }
      }
}
```

Dizi kopyalama

Var olan bir diziyi başka bir diziye kopyalamak için elemanları tek tek çekerek alabiliriz. Ancak bunun yerine aşağıdaki gibi bir yöntem bulunmaktadır. Bu yöntemde Java içerisindeki System sınıfının arraycopy metodu kullanılır. Metodun ilk parametresi içeriği alınacak dizi, ikincisi başlangıç notası, üçüncüsü kopyalama yapılacak dizi, dördüncüsü kopyalama yapılacak dizinin başlangıç noktası ve dördüncüsü ise kopyalama yapılacak dizinin kopyalama uzunluğudur.

```
public class Ornek {
      public static void main(String[] args) {
            String[] sehirler = new String[4];
            sehirler[0] = "Ankara";
            sehirler[1] = "Istanbul";
            sehirler[2] = "İzmir";
            sehirler[3] = "Balıkesir";

            String[] secilenSehirler = new String[4];

            System.arraycopy(sehirler, 2, secilenSehirler, 0, 2);
```

```
            for (int i = 0; i < secilenSehirler.length; i++) {
                  System.out.println(secilenSehirler[i]);
            }
      }
}
```

Çok boyutlu diziler

Yine bir çok programlama dilinde olduğu gibi Java' da da birden fazla boyutlu dizi bulunmaktadır. Aşağıdaki örneklerde birden fazla boyutlu dizi tanımları bulunmaktadır.

```
public class Ornek {
      public static void main(String[] args) {
            String[][] ikiBoyutluDizi = new String[4][2];
            int[][][] ucBoyutluDizi = new int[2][8][4];
            Ogrenci[][][][] dortBoyutluDizi = new Ogrenci[2][6][4][6];
      }
}
```

Çok boyutlu dizi elemanlarına erişim ve değer ataması yapmak için aşağıdaki yöntem kullanılabilir.

```
public class Ornek {
      public static void main(String[] args) {
            String[][] sehirler = new String[3][2];
            sehirler[0][0] = "İstanbul";
            sehirler[0][1] = "Edirne";
            sehirler[1][0] = "İzmir";
            sehirler[1][1] = "Aydın";
            sehirler[2][0] = "Ankara";
            sehirler[2][1] = "Konya";
      }
}
```

Atamalarda alternatif olarak ilk tanımla birlikte aşağıdaki gibi çok boyutlu bir dizi oluşturulabilir.

```
public class Ornek {
      public static void main(String[] args) {
            String[][] sehirler = { {"Istanbul", "Edirne" },
                                    {"Izmir", "Aydın" },
                                    {"Ankara", "Konya" }
                                  };
      }
}
```

Peki çok boyutlu dizilerde değerleri ekrana yazdırmak istersek ne yapmalıyız ? Burada yine dizi boyutunu otomatik olarak alabiliriz. Bunun için aşağıdaki örnekteki gibi iç içe kullanılan dizide length değeri alınarak dizinin elemanları yazdırılabilir.

```
public class Ornek {
	public static void main(String[] args) {
		String[][] sehirler = new String[3][2];
		sehirler[0][0] = "İstanbul";
		sehirler[0][1] = "Edirne";
		sehirler[1][0] = "İzmir";
		sehirler[1][1] = "Aydın";
		sehirler[2][0] = "Ankara";
		sehirler[2][1] = "Konya";

		for (int i = 0; i < sehirler.length; i++) {
			for (int j = 0; j < sehirler[i].length; j++) {
				System.out.println(sehirler[i][j]);
			}
		}
	}
}
```

Dikkat : Yukarıdaki örnekte dizinin uzunluğunu alırken ilk döngüde sehirler.length ile alınıyor, diğer döngüde ise alt değer sehirler[i].length şeklinde geliyor. Bunun nedeni dizinin length i ilk uzunluğu verir.

Alt dizi uzunluklari için alt dizilere erişmek zorundasınız. Bunun nedeni dizilerin her zaman düzenli olmayabileceğidir. Buna bir örnek vermek gerekirse aşağıdaki iki boyutlu dizinin uzunluğu sizce nedir ?

```
String[][] sehirler = { { "Istanbul", "Edirne", "Tekirdağ" },
                        { "Izmir", "Aydın" }, { "Ankara" } };
```

Yukarıdaki örnekteki dizi uzunluklarını aşağıdaki gibi yazdıralım.

```
public class Ornek {
	public static void main(String[] args) {
		String[][] sehirler = { { "Istanbul", "Edirne", "Tekirdağ" },
                        { "Izmir", "Aydın" }, { "Ankara" } };

		System.out.println(sehirler.length);
		System.out.println(sehirler[0].length);
		System.out.println(sehirler[1].length);
		System.out.println(sehirler[2].length);

	}
}
```

Buna göre çıktı 3, 3, 2, 1 şeklindedir. Yani bu 3 e 2 şeklinde bir dizi değildir ve düzeli bir boyutu yoktur. Bu durumda her alt dizinin farklı boyutu olacaktır.

Dikkat : Çok boyutlu diziler aslında kendi içlerinde birer alt dizi taşırlar. Buna örnek vermek gerekirse "int[3][4] ornekDizi" şeklindeki bir dizinin ornekDizi[0] ile ilk elemanını almaya çalıştığınızda bu bize 4 uzunluğunda tekil biri dizi verir.

Dikkat : Diziler için uzunluk artırma, kopyalama ve sıralama gibi operasyonlar için Collections Framewor bölümü içerisinde geçen Arrays sınıfını inceleyebilirsiniz.