Diziler (Arrays)

- Giriş
- Diziler
- Dizileri Tanımlama ve Yer Ayırma
- Dizi Kullanımına Yönelik Örnekler
  - Bir Diziye Yer Ayırma ve Elemanlarına İlk Değer Atama
  - Dizideki Elemanların Toplamı
  - Dizi Elemanlarının Sayaç Olarak Kullanılması
- Metodlara Dizi Gönderme
- Dizileri Değerle ve Referansla gönderme
- Dizilerin Sıralanması
- Dizilerde Arama: Doğrusal Arama (Linear Search) ve
- İkilik Arama (Binary Search)
  - Linear Search ile Arama
  - Binary Search ile Sıralı Dizilerde Arama
- Çok Boyutlu Diziler
- `for/each` Tekrar Yapısı

Giriş

- Veri Yapıları (Data structures)
  - Aynı türde olan bilgilerden oluşur
  - Statik yapıdadır. Birkez oluşturulduğunda sürekli aynı boyutta (size) kalır

Diziler

- Bir grup komşu hafıza alanı
  - Aynı isim
  - Aynı tip
- Dizi içindeki bir eleman sıra numarasıyla (*position number*) gösterilir
- Bir eleman dizinin adı ve parantez içinde ([]) (*position number, subscript*) elemanın pozisyon numarasıyla ifade edilir.
- İlk elemanın sıra numarası sıfırdır (*zeroth element)*
  - **c** dizinin ilk elemanı ve **c[0]** şeklinde gösterilir.

Diziler

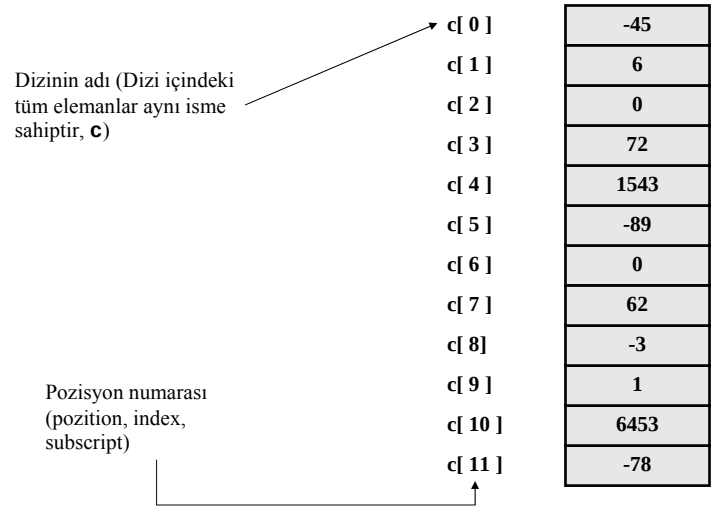


Fig. 7.1 12 elemanlı bir dizi

Dizileri Tanımlama ve Yer Ayırma

- Dizi içindeki elemanların türü programcı tarafından belirlenir
- **new** operatörü dizinin elemanlarını hafızaya yerleştirir
- Dizi tanımlama ve başlangıç değeri atama aynı deyim içinde olmayabilir
- Değer türündeki dizilerde her eleman belirtilen türde sadece bir değere sahiptir
- Referans türündeki dizilerde her eleman bir nesneyi gösterir

Dizileri Tanımlama ve Yer Ayırma

- Diziler **new** anahtar kelimesiyle kaç tane elemanı olduğu belirlenerek hafızaya yerleştirilir
- Diziler başlangıç listesiyle (*initializer lists*) ilk değerini alabilir
  - Diziye alan ayrılırken başlangıç listesindeki eleman sayısı kullanılır
  - Başlangıç listesindeki değerler ile dizinin elemanlarına değerleri atanır

```
1    // Fig 7.3: InitArray.cs
2    // Different ways of initializing arrays.
3
4    using System;
5    using System.Windows.Forms;
6
7    class InitArray
8    {
9       // main entry point for application
10      static void Main( string[] args )
11      {
12         string output = "";
13
14         int[] x;           // declare reference to an array
15         x = new int[ 10 ];  // dynamically allocate array and set
16                            // default values
17
18         // initializer list specifies number of elements
19         // and value of each element
20         int[] y = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 };
21
22         const int ARRAY_SIZE = 10; // named consta
23         int[] z;                   // reference to
24
25         // allocate array of ARRAY_SIZE (i.e., 10) elements
26         z = new int[ ARRAY_SIZE ];
27
28         // set the values in the array
29         for ( int i = 0; i < z.Length; i++ )
30            z[ i ] = 2 + 2 * i;
31
32         output += "Subscript\tArray x\tArray y\tArray z\n";
33
```

x, 10 boyutunda yerleştirilir

s dizisi integer olarak tanımlanır

ARRAY_SIZE adında sabit tanımlandı

Integer dizisi olarak y tanımandı ve başlangıç değerleri atandı

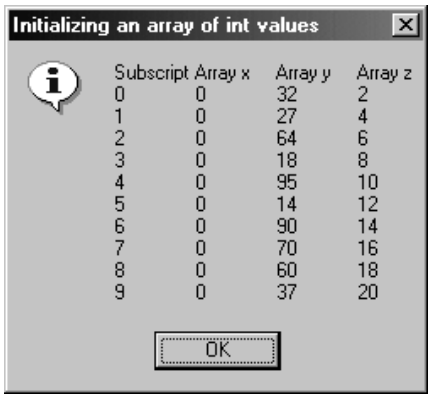
z dizisinin elemanları for döngüsüyle başlatıldı

z integer dizisi tanımlandı

z dizisi ARRAY_SIZE boyutunda başlatıldı

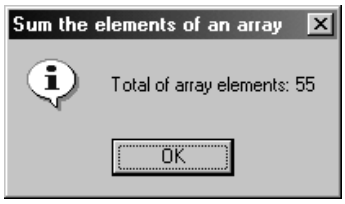
```
34         // output values for each array
35         for ( int i = 0; i < ARRAY_SIZE; i++ )
36            output += i + "\t" + x[ i ] + "\t" + y[ i ] +
37               "\t" + z[ i ] + "\n";
38
39         MessageBox.Show( output,
40            "Initializing an array of int values",
41            MessageBoxButtons.OK, MessageBoxIcon.Information );
42
43      } // end Main
44
45   } // end class InitArray
```

Dizilerdeki değerler output değişkenine ekleniyor



```
1    // Fig. 7.4: SumArray.cs
2    // Computing the sum of the elements in an array.
3
4    using System;
5    using System.Windows.Forms;
6
7    class SumArray
8    {
9       // main entry point for application
10      static void Main( string[] args )
11      {
12         int[] a = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
13         int total = 0;
14
15         for ( int i = 0; i < a.Length; i++ )
16            total += a[ i ];
17
18         MessageBox.Show( "Total of array elements: " + total,
19            "Sum the elements of an array",
20            MessageBoxButtons.OK, MessageBoxIcon.Information );
21
22      } // end Main
23
24   } // end class SumArray
```

a dizisinin toplam değeri hesaplanıyor




```
1    // Fig. 7.5: Histogram.cs
2    // Using data to create a histogram.
3
4    using System;
5    using System.Windows.Forms;
6
7    class Histogram
8    {
9       // main entry point for application
10      static void Main( string[] args )
11      {
12         int[] n = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
13         string output = "";
14
15         output += "Element\tvalue\tHistogram\n";
16
17         // build output
18         for ( int i = 0; i < n.Length; i++ )
19         {
20            output += "\n" + i + "\t" + n[ i ] + "\t";
21
22            for ( int j = 1; j <= n[ i ]; j++ ) // print a bar
23               output += "*";
24         }
25
26         MessageBox.Show( output, "Histogram Printing Program",
27            MessageBoxButtons.OK, MessageBoxIcon.Information );
28
29      } // end Main
30
31   } // end class Histogram
```

Integer olarak n dizisi tanımandı ve ilk değerler atandı

Dizideki her eleman için bar hazırlanıyor

Dizideki ilgili elemanın değerine bağlı olarak yıldızlarla bar oluşturuluyor

Dizi Elemanlarının Sayaç Olarak Kullanılması

- Dizi elemanları işlem sayılarının izlenmesinde kullanılabilir

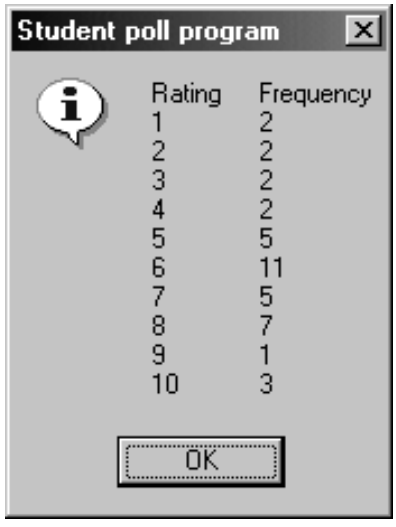
```
1    // Fig. 7.7: StudentPoll.cs
2    // A student poll program.
3
4    using System;
5    using System.Windows.Forms;
6
7    class StudentPoll
8    {
9       // main entry point for application
10      static void Main( string[] args )
11      {
12         int[] responses = { 1, 2, 6, 4, 8, 5, 9, 7, 8, 1
13            6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6,
14            5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10 };
15
16         int[] frequency = new int[ 11 ];
17         string output = "";
18
19         // increment the frequency for each response
20         for ( int answer = 0; answer < responses.Length; answer++ )
21            ++frequency[ responses[ answer ] ];
22
23         output += "Rating\tFrequency\n";
24
25         // output results
26         for ( int rating = 1; rating < frequency.Length; rating++ )
27            output += rating + "\t" + frequency[ rating ] + "\n";
28
29         MessageBox.Show( output, "Student poll program",
30            MessageBoxButtons.OK, MessageBoxIcon.Information );
31
32      } // end method Main
33
34   } // end class StudentPoll
```

Responses integer dizisi tanımlanarak başlatıldı

frequency integer dizisi tanımlandı ve yer ayrıldı

response dizisindeki her eleman için frequency dizisinde ilgili elemanın answer değerine göre artırılır

Her response değerinin kaç kez tekrarlandığı yazılır



Dizilerin Metodlara Gönderilmesi

- Dizi adıyla metoda parametre olarak gönderilir (Boyutunu gösteren parantez kullanılmaz)
- Diziler metoda referans olarak gönderilir
- Sadece dizinin bir elemanının değeri gönderilebilir

```
1    // Fig. 7.8: PassArray.cs
2    // Passing arrays and individual elements to methods.
3    using System;
4    using System.Drawing;
5    using System.Collections;
6    using System.ComponentModel;
7    using System.Windows.Forms;
8    using System.Data;
9
10   public class PassArray : System.Windows.Forms.Form
11   {
12      private System.Windows.Forms.Button showOutputButton;
13      private System.Windows.Forms.Label outputLabel;
14
15      // Visual Studio .NET generated code
16
17      [STAThread]
18      static void Main()
19      {
20         Application.Run( new PassArray() );
21      }
22
23      private void showOutputButton_Click( object sender,
24         System.EventArgs e )
25      {
26         int[] a = { 1, 2, 3, 4, 5 };
27
28         outputLabel.Text = "Effects of passing entire array " +
29            "call-by-reference:\n\nThe values of the original " +
30            "array are:\n\t";
31
32         for ( int i = 0; i < a.Length; i++ )
33            outputLabel.Text += "   " + a[ i ];
34
35         ModifyArray( a );   // array is passed by reference
```

A integer dizisi tanımlandı ve başlatıldı

ModifyArray metoduna a dizisi referans olarak gönderildi

Çıktı içeriği hazırlanıyor

```
36
37          outputLabel.Text +=
38             "\n\nThe values of the modified array are:\n\t";
39
40          // display elements of array a
41          for ( int i = 0; i < a.Length; i++ )
42             outputLabel.Text += "   " + a[ i ];
43
44          outputLabel.Text += "\n\nEffects of passing array
45             "element call-by-value:\n\na[ 3 ] before " +
46             "ModifyElement: " + a[ 3 ];
47
48          // array element passed call-by-value
49          ModifyElement( a[ 3 ] );
50
51          outputLabel.Text +=
52             "\na[ 3 ] after ModifyElement: " + a[ 3 ];
53       }
54
55       // method modifies the array it receives,
56       // original will be modified
57       public void ModifyArray( int[] b )
58       {
59          for ( int j = 0; j < b.Length; j++ )
60             b[ j ] *= 2;
61       }
62
63       // method modifies the integer passed to it
64       // original will not be modified
65       public void ModifyElement( int e )
66       {
67          outputLabel.Text +=
68             "\nvalue received in ModifyElement: " + e;
69
```

Her eleman kendi değerinin iki katıyla değiştiriliyor

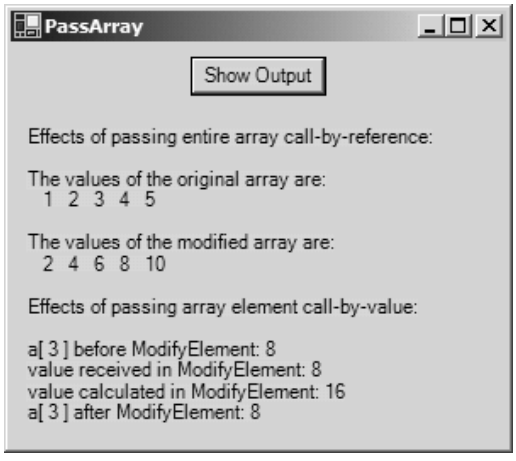
ModifyArray metodunun sonucundaki değerler yazdırılıyor

ModifyElement metodu 3. sıradaki eleman gönderilerek çalıştırılıyor

```
70          e *= 2;
71
72          outputLabel.Text +=
73             "\nvalue calculated in ModifyElement: " + e;
74       }
75    }
```

Değer iki ile çarpılıyor

Orijinal değişkenin değeri değişmez çünkü sadece elemanın değeri gönderildi



Dizilerin Değer ile ve Referans ile Gönderilmesi

- Bir nesneyi (object) saklayan değişkenler gerçekte nesnenin referansını saklar
- Bir referans bilgisayar hafızasında nesnenin kendisinin saklandığı yerdir
- Metodlara değer gönderme
  - Değişkenin bir kopyası çıkarılır
  - Yeni değişkendeki herhangi bir değişiklik orijinal değişkeni etkilemez
- Metodlara referans gönderme
  - Nesnenin bir kopyası çıkarılır
  - Nesnenin içeriğindeki değişiklik orijinal nesneyi etkiler

Dizilerin Değer ile ve Referans ile Gönderilmesi

- **ref** anahtar kelimesi metodlara referans ile gönderir
  - Değer tipindeki değişkenlerin metod içinde değerleri değiştiğinde orijinal değişkenin değeride değişir
  - Referans ile gönderilen nesnelerin referansı değiştiğinde orijinal referansta değişir

```
1    // Fig. 7.9: ArrayReferenceTest.cs
2    // Testing the effects of passing array references
3    // by value and by reference.
4    using System;
5    using System.Drawing;
6    using System.Collections;
7    using System.ComponentModel;
8    using System.Windows.Forms;
9    using System.Data;
10
11   public class ArrayReferenceTest : System.Windows.Forms.Form
12   {
13      private System.Windows.Forms.Label outputLabel;
14      private System.Windows.Forms.Button showOutputButton;
15
16      [STAThread]
17      static void Main()
18      {
19         Application.Run( new ArrayReferenceTest() );
20      }
21
22      private void showOutputButton_Click( object sender,
23         System.EventArgs e )
24      {
25         // create and initialize firstArray
26         int[] firstArray = { 1, 2, 3 };
27
28         // copy firstArray reference
29         int[] firstArrayCopy = firstArray;
30
31         outputLabel.Text +=
32            "Test passing firstArray reference by value";
33
34         outputLabel.Text += "\n\nContents of firstArray " +
35            "before calling FirstDouble:\n\t";
```

firstArray dizisi tanımlandı ve başlatıldı

firstArrayCopy dizisi firstArray dizisinin referansına sahip olarak kopyalandı

```
36
37         // print contents of firstArray
38         for ( int i = 0; i < firstArray.Length; i++ )
39            outputLabel.Text += firstArray[ i ] + " ";
40
41         // pass reference firstArray by value to FirstDouble
42         FirstDouble( firstArray );
43
44         outputLabel.Text += "\n\nContents of f
45            "calling FirstDouble\n\t";
46
47         // print contents of firstArray
48         for ( int i = 0; i < firstArray.Length; i++ )
49            outputLabel.Text += firstArray[ i ] + " ";
50
51         // test whether reference was changed b
52         if ( firstArray = = firstArrayCopy )
53            outputLabel.Text +=
54               "\n\nThe references refer to the same array\n";
55         else
56            outputLabel.Text +=
57               "\n\nThe references refer to different arrays\n";
58
59         // create and initialize secondArray
60         int[] secondArray = { 1, 2, 3 };
61
62         // copy secondArray reference
63         int[] secondArrayCopy = secondArray;
64
65         outputLabel.Text += "\nTest passing secondArray " +
66            "reference by reference";
67
68         outputLabel.Text += "\n\nContents of secondArray " +
69            "before calling SecondDouble:\n\t";
70
```

firstArray dizisinin çıktısı

FirstDouble metodu firstArray dizisiyle çağrıldı

firstArray ve firstArrayCopy aynı nesneyi referans ediyormu ?

secondArray dizisi tanımlandı ve başlatıldı

secondArrayCopy dizisi secondArray dizisinin referansına sahip olarak kopyalandı

firstArray dizisinin çıktısı

```
71          // print contents of secondArray before method call
72          for ( int i = 0; i < secondArray.Length; i++ )
73             outputLabel.Text += secondArray[ i ] + " ";
74
75          SecondDouble( ref secondArray );
76
77          outputLabel.Text += "\n\nContents of secondArray " +
78             "after calling SecondDouble:\n\t";
79
80          // print contents of secondArray after method call
81          for ( int i = 0; i < secondArray.Length; i++ )
82             outputLabel.Text += secondArray[ i ] + " ";
83
84          // test whether reference was changed by SecondDouble
85          if ( secondArray = = secondArrayCopy )
86             outputLabel.Text +=
87                "\n\nThe references refer to the same array\n";
88          else
89             outputLabel.Text +=
90                "\n\nThe references refer to different arrays\n"
91
92       } // end method showOutputButton_Click
93
94       // modify elements of array and attempt to modify
95       // reference
96       void FirstDouble( int[] array )
97       {
98          // double each element's value
99          for ( int i = 0; i < array.Length; i++ )
100            array[ i ] *= 2;
101
102         // create new reference and assign it to array
103         array = new int[] { 11, 12, 13 };
104      }
105
```

secondArray ve secondArrayCopy aynı nesneyi referans ediyormu ?

secondArray dizisinin çıktısı

array dizisi 11, 12ve 13 değerlerine sahip yeni bir dizisiyi referans etsin (?)

SecondDouble metodu SecondArray dizisiyle çağrıldı

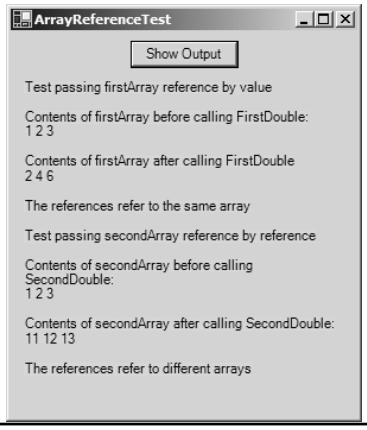
secondArray dizisinin çıktısı

Her eleman ikiyle çarpılır

```
106      // modify elements of array and change reference array
107      // to refer to a new array
108      void SecondDouble( ref int[] array )
109      {
110         // double each element's value
111         for ( int i = 0; i < array.Length; i++ )
112            array[ i ] *= 2;
113
114         // create new reference and assign it to array
115         array = new int[] { 11, 12, 13 };
116      }
117   }
```

Her eleman ikiyle çarpılır

array dizisi 11, 12ve 13 değerlerine sahip yeni bir dizisiyi referans etsin



Dizilerin Sıralanması

- Birçok uygulama için sıralama önemlidir
- Bubble Sort
  - Dizinin tümünde n (n eleman sayısı) adet geçiş yapar
  - Her geçişte, yan yana iki elemanı karşılaştırır
    - Eğer birinci ikinciden büyükse yer değiştirir
  - Programlama kolaydır
  - Diğer algoritmalara göre daha yavaştır
- .NET Framework çok hızlı sıralama (`Array.Sort( )`) ve arama (`Array.BinarySearch( )`) metodlarına sahiptir

```
1    // Fig. 7.10: BubbleSorter.cs
2    // Sorting an array's values into ascending order.
3    using System;
4    using System.Drawing;
5    using System.Collections;
6    using System.ComponentModel;
7    using System.Windows.Forms;
8    using System.Data;
9
10   public class BubbleSorter : System.Windows.Forms.Form
11   {
12      private System.Windows.Forms.Button sortButton;
13      private System.Windows.Forms.Label outputLabel;
14
15      // Visual Studio .NET generated code
16
17      [STAThread]
18      static void Main()
19      {
20         Application.Run( new BubbleSorter() );
21      }
22
23      private void sortButton_Click( object sender,
24            System.EventArgs e )
25      {
26         int[] a =  { 2, 6, 4, 8, 10, 12, 89, 68, 45, 37 };
27
28         outputLabel.Text += "data items in original order\n";
29
30         for ( int i = 0; i < a.Length; i++ )
31            outputLabel.Text += "   " + a[ i ];
32
33         // sort elements in array a
34         BubbleSort( a );
35
```

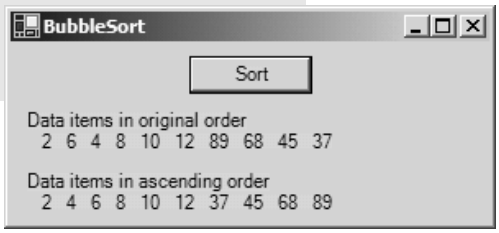
a dizisi BubbleSort metoduna gönderildi

```
36          outputLabel.Text += "\n\nData items in ascending order\n";
37
38          for ( int i = 0; i < a.Length; i++ )
39             outputLabel.Text += "   " + a[ i ];
40
41       } // end method sortButton_Click
42
43       // sort the elements of an array with bubble sort
44       public void BubbleSort( int[] b )
45       {
46          for ( int pass = 1; pass < b.Length; pass++ ) // passes
47
48             for ( int i = 0; i < b.Length - 1; i++ )
49
50                if ( b[ i ] > b[ i + 1 ] )        // one
51                   Swap( b, i );                  // one swap
52       }
53
54       // swap two elements of an array
55       public void Swap( int[] c, int first )
56       {
57          int hold;      // temporary holding area for swap
58
59          hold = c[ first ];
60          c[ first ] = c[ first + 1 ];
61          c[ first + 1 ] = hold;
62       }
63    }
```

Öndeki eleman ardından gelenden büyükse yer değiştirir

Dizideki iki eleman yer değiştirir

b dizisinin her elemanı için tekrar eder



Dizilerde Arama: Doğrusal Arama (Linear Search) ve İkilik Arama (Binary Search)

- Diziler çok büyük boyutlu olabilir
- Bir elemanın dizide olup olmadığı bilinmek istenebilir
- Linear Search
- Binary Search

Doğrusal Arama ile Dizlerde Arama

- Aranan anahtarın varsa dizideki sırasını döndürür
- Arama dizinin başından başlar ve sırayla devam eder
- Ortalama durumda (Average Case), istenen elemanın aranmasında dizisinin yarısını aramak gerekir
- Küçük ve sırasız dizilerde iyi çalışır

```
1    // Fig. 7.11: LinearSearcher.cs
2    // Demonstrating linear searching of an array.
3    using System;
4    using System.Drawing;
5    using System.Collections;
6    using System.ComponentModel;
7    using System.Windows.Forms;
8    using System.Data;
9
10   public class LinearSearcher : System.Windows.Forms.Form
11   {
12      private System.Windows.Forms.Button searchButton;
13      private System.Windows.Forms.TextBox inputTextBox;
14      private System.Windows.Forms.Label outputLabel;
15
16      int[] a = { 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26,
17                  28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50 };
18
19      // Visual Studio .NET generated code
20
21      [STAThread]
22      static void Main()
23      {
24         Application.Run( new LinearSearcher() );
25      }
26
27      private void searchButton_Click( object sender,
28         System.EventArgs e )
29      {
30         int searchKey = Int32.Parse( inputTextBox.Text );
31
32         int elementIndex = LinearSearch( a, searchKey );
33
```

Kullanıcıdan aranacak anahtarı alır

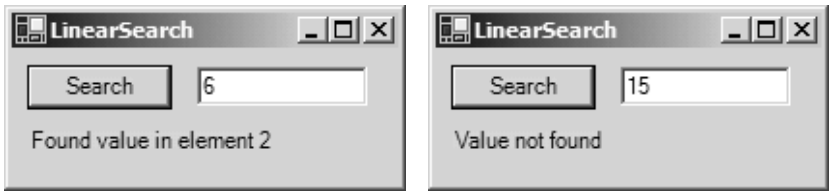
Doğrusal aramayla anahtarı arar

```
34          if ( elementIndex != -1 )
35             outputLabel.Text =
36                "Found value in element " + elementIndex;
37
38          else
39             outputLabel.Text = "Value not found";
40
41       } // end method searchButton_Click
42
43       // search array for the specified key value
44       public int LinearSearch( int[] array, int key )
45       {
46          for ( int n = 0; n < array.Length; n++ )
47          {
48             if ( array[ n ] == key )
49                return n;
50          }
51
52          return -1;
53
54       } // end method LinearSearch
55
56    } // end class LinearSearcher
```

Eğer anahtarın indeksi -1 dönerse eleman dizide yoktur

Arama elelman yoksa -1 ile sonuçlanır

Dizinin başından başlanır ve tüm elemanlar sırayla kontrol edilir. Eğer eleman bulunursa sırası (index) geri döndürülür



Sıralı Dizide Binary Search

- Dizi sıralı olmak zorundadır
- Her adımda dizinin yarısı elimine edilir
- Algoritma
  - Ortadaki elemanı bul
  - Aranan anahtarla karşılaştır
    - Eğer eşitse eleman bulunmuştur ve sırasını döndür
    - Eğer anahtar küçükse aramaya dizinin ilk yarısıyla devam et
    - Eğer anahtar büyükse aramaya dizinin ikinci yarısıyla devam et
  - Yukarıdaki adımlar eleman bulununcaya kadar veya bir eleman kalıncaya kadar tekrarlanır

```
1    // Fig. 7.12: BinarySearchTest.cs
2    // Demonstrating a binary search of an array.
3
4    using System;
5    using System.Drawing;
6    using System.Collections;
7    using System.ComponentModel;
8    using System.Windows.Forms;
9    using System.Data;
10
11   public class BinarySearchTest : System.Windows.Forms.Form
12   {
13      private System.Windows.Forms.Label promptLabel;
14
15      private System.Windows.Forms.TextBox inputTextBox;
16
17      private System.Windows.Forms.Label resultLabel;
18      private System.Windows.Forms.Label displayLabel;
19      private System.Windows.Forms.Label outputLabel;
20
21      private System.Windows.Forms.Button findButton;
22
23      private System.ComponentModel.Container components = null;
24
25      int[] a = { 0, 2, 4, 6, 8, 10, 12, 14, 16,
26                  18, 20, 22, 24, 26, 28 };
27
28      // Visual Studio .NET generated code
29
30      // main entry point for application
31      [STAThread]
32      static void Main()
33      {
34         Application.Run( new BinarySearchTest() );
35      }
```

```
36
37      // searches for an element by calling
38      // BinarySearch and displaying results
39      private void findButton_Click( object sender,
40         System.EventArgs e )
41      {
42         int searchKey = Int32.Parse( inputTextBox.Text );
43
44         // initialize display string for the new sear
45         outputLabel.Text = "Portions of array searche
46
47         // perform the binary search
48         int element = BinarySearch( a, searchKey );
49
50         if ( element != -1 )
51            displayLabel.Text = "Found value in elem
52               element;
53         else
54            displayLabel.Text = "Value not found";
55
56      } // end findButton_Click
57
58      // searchs array for specified key
59      public int BinarySearch( int[] array, int key )
60      {
61         int low = 0;                  // low subscript
62         int high = array.Length - 1;  // high subscript
63         int middle;                   // middle subscript
64
65         while ( low <= high )
66         {
67            middle = ( low + high ) / 2;
68
```

Kullanıcıdan aranan anahtar alınır

BinarySearch metodu dizi ve istenen anahtarla çağrılır

Eğer –1 dönderse aranan eleman yoktur

Eğer low index high index değerinden küçükse aramaya devam eder

Kalan arama alanının ortasını bulur

```
69            // the following line displays the portion
70            // of the array currently being manipulated during
71            // each iteration of the binary search loop
72            BuildOutput( a, low, middle, high );
73
74            if ( key == array[ middle ] )   // match
75               return middle;
76            else if ( key < array[ middle ] )
77               high = middle - 1;   // search low end of array
78            else
79               low = middle + 1;
80
81         } // end BinarySearch
82
83         return -1;  // search key not found
84
85      } // end method BinarySearch
86
87      public void BuildOutput(
88         int[] array, int low, int mid, int high )
89      {
90         for ( int i = 0; i < array.Length; i++ )
91         {
92            if ( i < low || i > high )
93               outputLabel.Text += "    ";
94
95            // mark middle element in output
96            else if ( i = = mid )
97               outputLabel.Text +=
98                  array[ i ].ToString( "00" ) + "* ";
```

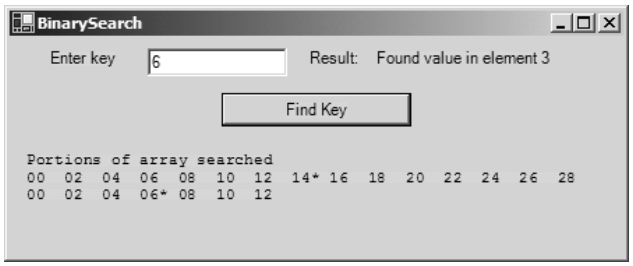
Ortadaki elemanı işaretleyerek kalan diznin tümünü yazar

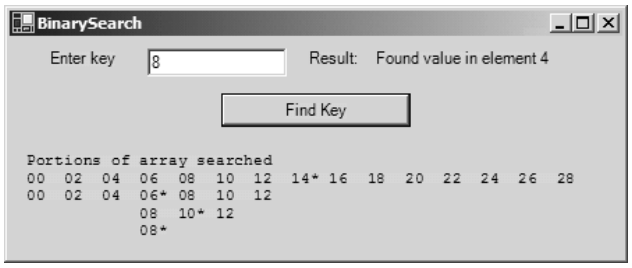
Eğer ortadaki eleman aranan anahtarla eşitse sırası döndürülür

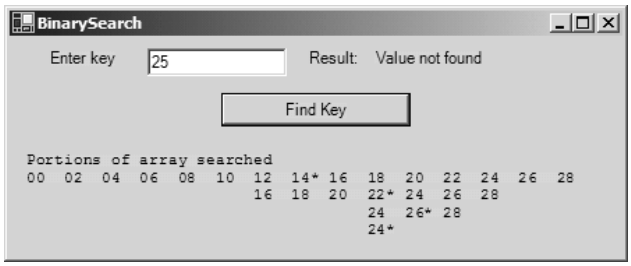
Eğer aranan değer ortadakinden küçükse hig indeksi ortadakinden bir eksik olarak al

Diğer durumda low indeksi ortadakinden bir fazla al

```
99            else
100              outputLabel.Text +=
101                 array[ i ].ToString( "00" ) + "  ";
102         }
103
104         outputLabel.Text += "\n";
105
106      } // end BuildOutput
107
108   } // end class BinarySearchTest
```







Çok Boyutlu Diziler

- Bir elemanın gösteriminde birden fazla indeks numarası kullanılır
- Dikdörtgen diziler
  - Genellikle tablolar gösterilir ve aynı boyutta sütun ve aynı boyutta satırdan oluşur
  - Elemanın ilk indeksi satırı ikinci indeksi sütunu gösterir
- Jagged Arrays
  - Dizilerin dizileri
  - Farklı boyuttaki dizilerden oluşan dizilerin oluşturduğu dizilerdir

Çok Boyutlu Diziler

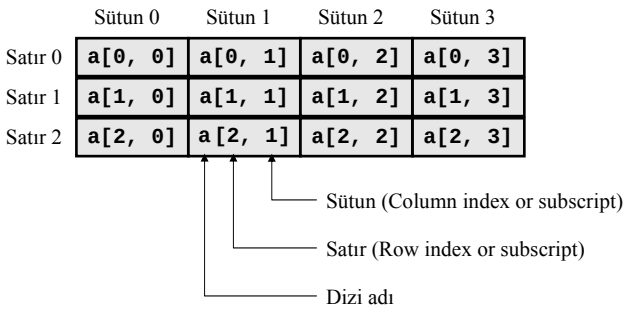


Fig. 7.13 Üç satır ve dört sütundan oluşan iki boyutlu dizi

```
1    // Fig. 7.14: TwoDimensionalArrays.cs
2    // Initializing two-dimensional arrays.
3    using System;
4    using System.Drawing;
5    using System.Collections;
6    using System.ComponentModel;
7    using System.Windows.Forms;
8    using System.Data;
9
10   public class TwoDimensionalArrays : System.Windows.Forms.Form
11   {
12      private System.Windows.Forms.Button show
13      private System.Windows.Forms.Label outpu
14
15      // Visual Studio .NET generated code
16
17      [STAThread]
18      static void Main()
19      {
20         Application.Run( new TwoDimensionalArrays() );
21      }
22
23      private void showOutputButton_Click( object sender,
24         System.EventArgs e )
25      {
26         // declaration and initialization of rectangular
27         int[,] array1 = new int[,] { { 1, 2, 3 }, { 4, 5, 6 } };
28
29         // declaration and initialization of
30         int[][] array2 = new int[ 3 ][];
31         array2[ 0 ] = new int[] { 1, 2 };
32         array2[ 1 ] = new int[] { 3 };
33         array2[ 2 ] = new int[] { 4, 5, 6 };
34
35         outputLabel.Text += "Values in array1 by row are\n";
```

array2 dizisinin ilk elemanı iki elemanlı bir dizi

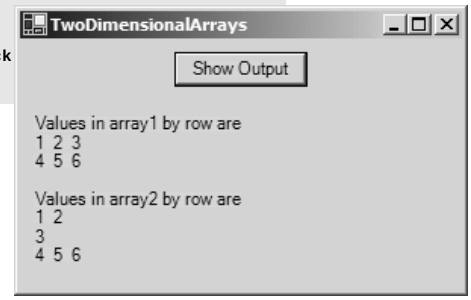
array2 dizisinin ikinci elemanı bir elemanlı bir dizi

3 satırlı jagged array tanımlandı

array1 adında iki boyutlu dizi tanımlandı ve başlatıldı

array2 dizisinin üçüncü elemanı üç elemanlı bir dizi

```
36
37          // output values in array1
38          for ( int i = 0; i < array1.GetLength( 0 ); i++ )
39          {
40             for ( int j = 0; j < array1.GetLength( 1 ); j++ )
41                outputLabel.Text += array1[ i, j ] + "  ";
42
43             outputLabel.Text += "\n";
44          }
45
46          outputLabel.Text += "\nValues in array2 by row are\n";
47
48          // output values in array2
49          for ( int i = 0; i < array2.Length; i++ )
50          {
51             for ( int j = 0; j < array2[ i ].Length; j++ )
52                outputLabel.Text += array2[ i ][ j ] + "  ";
53
54             outputLabel.Text += "\n";
55          }
56
57       } // end method showOutputButton_Click
58
59    } // end class TwoDimensionalArrays
```




```
1    // Fig. 7.15: DoubleArray.cs
2    // Manipulating a double-subscripted array.
3    using System;
4    using System.Drawing;
5    using System.Collections;
6    using System.ComponentModel;
7    using System.Windows.Forms;
8    using System.Data;
9
10   public class DoubleArray : System.Windows.Forms.Form
11   {
12      private System.Windows.Forms.Button showOutputButton;
13      private System.Windows.Forms.Label outputLabel;
14
15      int[][] grades;
16      int students, exams;
17
18      // Visual Studio .NET generated code
19
20      [STAThread]
21      static void Main()
22      {
23         Application.Run( new DoubleArray() );
24      }
25
26      private void showOutputButton_Click( object sender,
27         System.EventArgs e )
28
29      {
30         grades = new int[ 3 ][];
31         grades[ 0 ] =  new int[]{ 77, 68, 86, 73 };
32         grades[ 1 ] =  new int[]{ 96, 87, 89, 81 };
33         grades[ 2 ] =  new int[]{ 70, 90, 86, 81 };
34
```

Değişkenler global tanımlandı

grades dizisi üç satırla başlatıldı

grades dizisindeki her eleman (satır) başlatıldı

```
35         students = grades.Length;      // number of students
36         exams = grades[ 0 ].Length;    // number of exams
37
38         // line up column headings
39         outputLabel.Text += "                    ";
40
41         // output the column headings
42         for ( int i = 0; i < exams; i++ )
43            outputLabel.Text += "[" + i + "]  ";
44
45         // output the rows
46         for ( int i = 0; i < students; i++ )
47         {
48            outputLabel.Text += "\ngrades[" + i + "]   ";
49
50            for ( int j = 0; j < exams; j++ )
51               outputLabel.Text += grades[ i ][ j ] + "   ";
52         }
53
54         outputLabel.Text += "\n\nLowest grade: " + Minimum() +
55            "\nHighest grade: " + Maximum() + "\n";
56
57         for ( int i = 0; i < students; i++ )
58            outputLabel.Text += "\nAverage for student " + i + " is " +
59               Average( grades[ i ] );
60
61      } // end method showOutputButton_Click
62
```

Her satır yazıldı

Her satırdaki her eleman yazıldı

Minimum ve maksimum değerler yazıldı

Her satrı için average değer yazıldı

```
63      // find minimum grade in grades array
64      public int Minimum()
65      {
66         int lowGrade = 100;
67
68         for ( int i = 0; i < students; i++ )
69
70            for ( int j = 0; j < exams; j++ )
71
72               if ( grades[ i ][ j ] < lowGrade )
73                  lowGrade = grades[ i ][ j ];
74
75         return lowGrade;
76      }
77
78      // find maximum grade in grades array
79      public int Maximum()
80      {
81         int highGrade = 0;
82
83         for ( int i = 0; i < students; i++ )
84
85            for ( int j = 0; j < exams; j++ )
86
87               if ( grades[ i ][ j ] > highGrade )
88                  highGrade = grades[ i ][ j ];
89
90         return highGrade;
91      }
92
```

Dizideki her elemana bakılır

Eğer şimdiki eleman highGrade değerinden küçükse highGrade değişkeninin yeni değeri yap

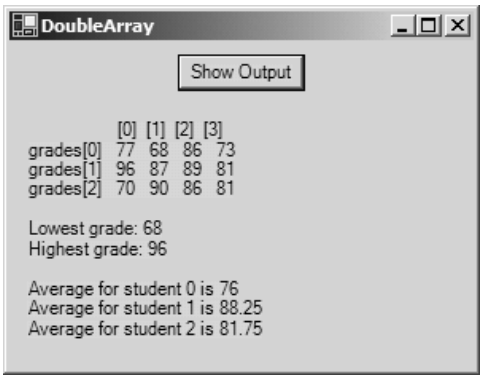
Dizideki her elemana bakılır

Eğer şimdiki eleman lowGrade değerinden küçükse lowGrade değişkeninin yeni değeri yap

```
93      // determine average grade for a particular student
94      public double Average( int[] setOfGrades )
95      {
96         int total = 0;
97
98         for ( int i = 0; i < setOfGrades.Length; i++ )
99            total += setOfGrades[ i ];
100
101        return ( double ) total / setOfGrades.Length;
102     }
103
104  } // end class DoubleArray
```

Dizideki toplam grade değeri

Toplam grade değerini grade sayısına böl



foreach tekrar yapısı

- foreach tekrar yapısı diziler gibi veri yapılarında tüm elemanlar için iterasyon yapmak için kullanılır
- Sayaç gerekmez
- Herbir elemanı göstermek için bir değişken kullanılır

```
1    // Fig. 7.16: ForEach.cs
2    // Demonstrating for/each structure.
3    using System;
4
5    class ForEach
6    {
7       // main entry point for the application
8       static void Main( string[] args )
9       {
10         int[,] gradeArray = { { 77, 68, 86, 73 },
11            { 98, 87, 89, 81 }, { 70, 90, 86, 81 } };
12
13         int lowGrade = 100;
14
15         foreach ( int grade in gradeArray )
16         {
17            if ( grade < lowGrade )
18               lowGrade = grade;
19         }
20
21         Console.WriteLine( "The minimum grade is: " + lowGrade );
22      }
23   }
```

foreach döngüsüyle dizideki tüm elemanlar taranır

Eğer dizideki şimdiki eleman lowGrade değerinden küçüksek lowGrade değerine atılır

```
The minimum grade is: 68
```

Haftalık Ödev

Bir sınıftaki 30 öğrencinin “Numarası”, “Adı ve Soyadı” ile “C#.NET”, “Veri Yapıları”, “Algoritma Analizi” adlı üç derse ait not bilgilerini kaydeden ve aşağıdaki işlemleri yapan bir programı metod ve dizi yapılarını kullanarak yazınız.

– Öğrenci bilgileri (numara, ad ve soyad) ve ders notları ((vize1 (%25), vize2(%25), final(%50)) kayıt

– Seçilen bir ders için
- sınıf ortalamasının üzerindeki öğrencilerin listesi
- sınıf ortalamasının altındaki öğrencilerin listesi
- girilen bir nottan yüksek veya düşük ortalaması olan alan öğrencilerin listesi

– Tüm öğrencilerin istenen bir derse göre not ortalaması büyükten küçüğe veya küçükten büyüğe doğru sıralanmış listesi

– Tüm öğrencilerin üç dersin toplam not ortalamasına göre büyükten küçüğe ve küçükten büyüğe doğru sıralaması

Not:

Ödev program çıktısı şeklinde teslim edilecek. Kapak sayfası örneği http://w3.gazi.edu.tr/web/akcayol adresinden alınabilir. Ödev başlığı olarak “ÖĞRENCİ OTOMASYON PROGRAMI” yazılacaktır.