Chapter 6 Sample Programs

```
//********************************************************************
//  BasicArray.java       Author: Lewis and Loftus
//  Demonstrates basic array declaration and use.
//********************************************************************
public class BasicArray
{
   final static int LIMIT = 15;
   final static int MULTIPLE = 10;
   //-----------------------------------------------------------------
   //  Creates an array, fills it with various integer values,
   //  modifies one value, then prints them out.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      int[] list = new int[LIMIT];

      //  Initialize the array values
      for (int index = 0; index < LIMIT; index++)
         list[index] = index * MULTIPLE;

      list[5] = 999;  // change one array value
      for (int index = 0; index < LIMIT; index++)
         System.out.print (list[index] + "  ");
      System.out.println ();
   }
}

//********************************************************************
//  ReverseNumbers.java       Author: Lewis and Loftus
//  Demonstrates array index processing.
//********************************************************************
import cs1.Keyboard;
public class ReverseNumbers
{
   //-----------------------------------------------------------------
   //  Reads a list of numbers from the user, storing them in an
   //  array, then prints them in the opposite order.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      double[] numbers = new double[10];
      System.out.println ("The size of the array: " + numbers.length);
      for (int index = 0; index < numbers.length; index++)
      {
         System.out.print ("Enter number " + (index+1) + ": ");
         numbers[index] = Keyboard.readDouble();
      }
      System.out.println ("The numbers in reverse:");
      for (int index = numbers.length-1; index >= 0; index--)
         System.out.print (numbers[index] + "  ");
      System.out.println ();
   }
}
```

```
//********************************************************************
//  LetterCount.java       Author: Lewis and Loftus
//  Demonstrates the relationship between arrays and strings.
//********************************************************************
import cs1.Keyboard;
public class LetterCount
{
   //-----------------------------------------------------------------
   //  Reads a sentence from the user and counts the number of
   //  uppercase and lowercase letters contained in it.
   //-----------------------------------------------------------------

   public static void main (String[] args)
   {
      final int NUMCHARS = 26;

      int[] upper = new int[NUMCHARS];
      int[] lower = new int[NUMCHARS];

      char current;   // the current character being processed
      int other = 0;  // counter for non-alphabetics

      System.out.println ("Enter a sentence:");
      String line = Keyboard.readString();

      //  Count the number of each letter occurance
      for (int ch = 0; ch < line.length(); ch++)
      {
         current = line.charAt(ch);
         if (current >= 'A' && current <= 'Z')
            upper[current-'A']++;
         else
            if (current >= 'a' && current <= 'z')
               lower[current-'a']++;
            else
               other++;
      }

      //  Print the results
      System.out.println ();
      for (int letter=0; letter < upper.length; letter++)
      {
         System.out.print ( (char) (letter + 'A') );
         System.out.print (": " + upper[letter]);
         System.out.print ("\t\t" + (char) (letter + 'a') );
         System.out.println (": " + lower[letter]);
      }

      System.out.println ();
      System.out.println ("Non-alphabetic characters: " + other);
   }
}
```

```
//********************************************************************
//  Primes.java       Author: Lewis and Loftus
//  Demonstrates the use of an initializer list for an array.
//********************************************************************

public class Primes
{
   //-----------------------------------------------------------------
   //  Stores some prime numbers in an array and prints them.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      int[] primes = {2, 3, 5, 7, 11, 13, 17, 19};

      System.out.println ("Array length: " + primes.length);

      System.out.println ("The first few prime numbers are:");

      for (int prime = 0; prime < primes.length; prime++)
         System.out.print (primes[prime] + "  ");

      System.out.println ();
   }
}

//********************************************************************
//  GradeRange.java       Author: Lewis and Loftus
//  Demonstrates the use of an array of String objects.
//********************************************************************

public class GradeRange
{
   //-----------------------------------------------------------------
   //  Stores the possible grades and their numeric lowest value,
   //  then prints them out.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      String[] grades = {"A", "A-", "B+", "B", "B-", "C+", "C", "C-",
                         "D+", "D", "D-", "F"};

      int[] cutoff = {95, 90, 87, 83, 80, 77, 73, 70, 67, 63, 60, 0};

      for (int level = 0; level < cutoff.length; level++)
         System.out.println (grades[level] + "\t" + cutoff[level]);
   }
}
```

```
//********************************************************************
//  NameTag.java       Author: Lewis and Loftus
//  Demonstrates the use of command line arguments.
//********************************************************************

public class NameTag
{
   //-----------------------------------------------------------------
   //  Prints a simple name tag using a greeting and a name that is
   //  specified by the user.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      System.out.println ();
      System.out.println ("     " + args[0]);
      System.out.println ("My name is " + args[1]);
      System.out.println ();
   }
}
//********************************************************************
//  Tunes.java       Author: Lewis and Loftus
//  Driver for demonstrating the use of an array of objects.
//********************************************************************

import CDCollection;

public class Tunes
{
   //-----------------------------------------------------------------
   //  Creates a CDCollection object and adds some CDs to it. Prints
   //  reports on the status of the collection.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      CDCollection music = new CDCollection ();

      music.addCD ("Storm Front", "Billy Joel", 14.95, 10);
      music.addCD ("Come On Over", "Shania Twain", 14.95, 16);
      music.addCD ("Soundtrack", "Les Miserables", 17.95, 33);
      music.addCD ("Graceland", "Paul Simon", 13.90, 11);

      System.out.println (music);

      music.addCD ("Double Live", "Garth Brooks", 19.99, 26);
      music.addCD ("Greatest Hits", "Jimmy Buffet", 15.95, 13);

      System.out.println (music);
   }
}
```

```
//********************************************************************
//  CDCollection.java       Author: Lewis and Loftus
//  Represents a collection of compact discs.
//********************************************************************
import CD;
import java.text.NumberFormat;

public class CDCollection
{
   private CD[] collection;
   private int count;
   private double totalValue;
   private int currentSize;
   //-----------------------------------------------------------------
   //  Creates an initially empty collection.
   //-----------------------------------------------------------------
   public CDCollection ()
   {
      currentSize = 100;
      collection = new CD[currentSize];
      count = 0;
      totalValue = 0.0;
   }
   //-----------------------------------------------------------------
   //  Adds a CD to the collection, increasing the size of the
   //  collection if necessary.
   //-----------------------------------------------------------------
   public void addCD (String title, String artist, double value, int tracks)
   {
      if (count == currentSize)
         increaseSize();

      collection[count] = new CD (title, artist, value, tracks);
      totalValue += value;
      count++;
   }

   //-----------------------------------------------------------------
   //  Returns a report describing the CD collection.
   //-----------------------------------------------------------------
   public String toString()
   {
      NumberFormat fmt = NumberFormat.getCurrencyInstance();

      String report = "&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&\n";
      report += "My CD Collection\n\n";

      report += "Number of CDs: " + count + "\n";
      report += "Total value: " + fmt.format(totalValue) + "\n";
      report += "Average cost: " + fmt.format(totalValue/count);
      report += "\n\nCD List:\n\n";

      for (int cd = 0; cd < count; cd++)
         report += collection[cd].toString() + "\n";
      return report;
   }
```

```
   //-----------------------------------------------------------------
   //  Doubles the size of the collection by creating a larger array
   //  and copying into it the existing collection.
   //-----------------------------------------------------------------
   private void increaseSize ()
   {
      currentSize *= 2;

      CD[] temp = new CD[currentSize];

      for (int cd = 0; cd < collection.length; cd++)
         temp[cd] = collection[cd];

      collection = temp;
   }
}
//********************************************************************
//  CD.java       Author: Lewis and Loftus
//  Represents a compact disc.
//********************************************************************

import java.text.NumberFormat;

public class CD
{
   private String title, artist;
   private double value;
   private int tracks;

   //-----------------------------------------------------------------
   //  Creates a new CD with the specified information.
   //-----------------------------------------------------------------
   public CD (String theTitle, String theArtist, double theValue,
              int theTracks)
   {
      title = theTitle;
      artist = theArtist;
      value = theValue;
      tracks = theTracks;
   }

   //-----------------------------------------------------------------
   //  Returns a description of this CD.
   //-----------------------------------------------------------------
   public String toString()
   {
      NumberFormat fmt = NumberFormat.getCurrencyInstance();

      String description;

      description = fmt.format(value) + "\t" + tracks + "\t";
      description += title + "\t" + artist;

      return description;
   }
}
```

```
//********************************************************************
//  SortGrades.java       Author: Lewis and Loftus
//  Driver for testing a numeric selection sort.
//********************************************************************

import Sorts;

public class SortGrades
{
   //-----------------------------------------------------------------
   //  Creates an array of grades, sorts them, then prints them.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      int[] grades = {89, 94, 69, 80, 97, 85, 73, 91, 77, 85, 93};

      Sorts.selectionSort (grades);

      for (int index = 0; index < grades.length; index++)
         System.out.print (grades[index] + "   ");
   }
}
//********************************************************************
//  Sorts.java       Author: Lewis and Loftus
//  Demonstrates the selection sort and insertion sort algorithms,
//  as well as a generic object sort.
//********************************************************************

public class Sorts
{
   //-----------------------------------------------------------------
   //  Sorts the specified array of integers using the selection
   //  sort algorithm.
   //-----------------------------------------------------------------
   public static void selectionSort (int[] numbers)
   {
      int min, temp;

      for (int index = 0; index < numbers.length-1; index++)
      {
         min = index;
         for (int scan = index+1; scan < numbers.length; scan++)
            if (numbers[scan] < numbers[min])
               min = scan;

         // Swap the values
         temp = numbers[min];
         numbers[min] = numbers[index];
         numbers[index] = temp;
      }
   }
```

```
  //-----------------------------------------------------------------
  //  Sorts the specified array of integers using the insertion
  //  sort algorithm.
  //-----------------------------------------------------------------
  public static void insertionSort (int[] numbers)
  {
    for (int index = 1; index < numbers.length; index++)
    {
      int key = numbers[index];
      int position = index;

      // shift larger values to the right
      while (position > 0 && numbers[position-1] > key)
      {
        numbers[position] = numbers[position-1];
        position--;
      }

      numbers[position] = key;
    }
  }

  //-----------------------------------------------------------------
  //  Sorts the specified array of objects using the insertion
  //  sort algorithm.
  //-----------------------------------------------------------------
  public static void insertionSort (Comparable[] objects)
  {
    for (int index = 1; index < objects.length; index++)
    {
      Comparable key = objects[index];
      int position = index;

      // shift larger values to the right
      while (position > 0 && objects[position-1].compareTo(key) > 0)
      {
        objects[position] = objects[position-1];
        position--;
      }

      objects[position] = key;
    }
  }
}
```

```
//********************************************************************
//  SortPhoneList.java       Author: Lewis and Loftus
//  Driver for testing an object selection sort.
//********************************************************************
import Contact;
import Sorts;
class SortPhoneList
{
   //-----------------------------------------------------------------
   //  Creates an array of Contact objects, sorts them, then prints them.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      Contact[] friends = new Contact[7];
      friends[0] = new Contact ("John", "Smith", "610-555-7384");
      friends[1] = new Contact ("Sarah", "Barnes", "215-555-3827");
      friends[2] = new Contact ("Mark", "Riley", "733-555-2969");
      friends[3] = new Contact ("Laura", "Getz", "663-555-3984");
      friends[4] = new Contact ("Larry", "Smith", "464-555-3489");
      friends[5] = new Contact ("Frank", "Phelps", "322-555-2284");
      friends[6] = new Contact ("Marsha", "Grant", "243-555-2837");
      Sorts.insertionSort(friends);
      for (int index = 0; index < friends.length; index++)
         System.out.println (friends[index]);
   }
}

//********************************************************************
//  Contact.java       Author: Lewis and Loftus
//  Represents a phone contact.
//********************************************************************
class Contact implements Comparable
{
   private String firstName, lastName, phone;
   //-----------------------------------------------------------------
   //  Sets up this contact with the specified information.
   //-----------------------------------------------------------------
   public Contact (String firstName, String lastName, String phone)
   {
      this.firstName = firstName;
      this.lastName = lastName;
      this.phone = phone;
   }
   //-----------------------------------------------------------------
   //  Returns a description of this contact as a string.
   //-----------------------------------------------------------------
   public String toString ()
   {
      return lastName + ", " + firstName + "\t" + phone;
   }
```

```
   //-----------------------------------------------------------------
   //  Uses both last and first names to determine lexical ordering.
   //-----------------------------------------------------------------
   public int compareTo (Object other)
   {
      int result;
      if (lastName.equals(((Contact)other).lastName))
         result = firstName.compareTo(((Contact)other).firstName);
      else
         result = lastName.compareTo(((Contact)other).lastName);
      return result;
   }
}

//********************************************************************
// TwoDArray.java Author: Lewis and Loftus // // Demonstrates the use of a two-dimensional array.
//********************************************************************
public class TwoDArray
{
    //-----------------------------------------------------------------
    // Creates a 2D array of integers, fills it with increasing
    // integer values, then prints them out.
    //-----------------------------------------------------------------
    public static void main (String[] args)
    {
        int[][] table = new int[5][10];           // Load the table with values
        for (int row=0; row < table.length; row++)
              for (int col=0; col < table[row].length; col++)
                    table[row][col] = row * 10 + col;
        // Print the table
        for (int row=0; row < table.length; row++)
        {
              for (int col=0; col < table[row].length; col++)
                    System.out.print (table[row][col] + "\t"); System.out.println();
        }
    }
}
```

```
//********************************************************************
//  Beatles.java       Author: Lewis and Loftus
//  Demonstrates the use of a Vector object.
//********************************************************************
import java.util.Vector;

public class Beatles
{
   //-----------------------------------------------------------------
   //  Stores and modifies a list of band members.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      Vector band = new Vector();

      band.addElement ("Paul");
      band.addElement ("Pete");
      band.addElement ("John");
      band.addElement ("George");

      System.out.println (band);

      band.removeElement ("Pete");

      System.out.println (band);
      System.out.println ("At index 1: " + band.elementAt(1));

      band.insertElementAt ("Ringo", 2);

      System.out.println (band);
      System.out.println ("Size of the band: " + band.size());
   }
}

//********************************************************************
//  Rocket.java       Author: Lewis and Loftus
//  Demonstrates the use of polygons and polylines.
//********************************************************************

import java.applet.Applet;
import java.awt.*;

public class Rocket extends Applet
{
   private final int APPLET_WIDTH = 200;
   private final int APPLET_HEIGHT = 200;

   private int[] xRocket = {100, 120, 120, 130, 130, 70, 70, 80, 80};
   private int[] yRocket = {15, 40, 115, 125, 150, 150, 125, 115, 40};

   private int[] xWindow = {95, 105, 110, 90};
   private int[] yWindow = {45, 45, 70, 70};

   private int[] xFlame = {70, 70, 75, 80, 90, 100, 110, 115, 120, 130, 130};
   private int[] yFlame = {155, 170, 165, 190, 170, 175, 160, 185, 160, 175, 155};
```

```
   //-----------------------------------------------------------------
   //  Sets up the basic applet environment.
   //-----------------------------------------------------------------
   public void init()
   {
      setBackground (Color.black);
      setSize (APPLET_WIDTH, APPLET_HEIGHT);
   }

   //-----------------------------------------------------------------
   //  Draws a rocket using polygons.
   //-----------------------------------------------------------------
   public void paint (Graphics page)
   {
      page.setColor (Color.cyan);
      page.fillPolygon (xRocket, yRocket, xRocket.length);
      page.setColor (Color.gray);
      page.fillPolygon (xWindow, yWindow, xWindow.length);
      page.setColor (Color.red);
      page.drawPolyline (xFlame, yFlame, xFlame.length);
   }
}

<! Rocket.html>
<HTML>
<HEAD>
<TITLE>The Rocket Applet</TITLE>
</HEAD>
<BODY>
<center>
<H3>The Rocket Applet:</H3>
<APPLET CODE="Rocket.class" WIDTH=200 HEIGHT=200></APPLET>
<HR>
</center>
</BODY>
</HTML>

//********************************************************************
//  Dots2.java       Author: Lewis and Loftus
//  Demonstrates the use of a Vector to store the state of a drawing.
//********************************************************************

import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import java.util.*;

public class Dots2 extends Applet implements MouseListener
{
   private final int APPLET_WIDTH = 200;
   private final int APPLET_HEIGHT = 100;
   private final int RADIUS = 6;

   private Vector pointList;
   private int count;
```

```
   //-----------------------------------------------------------------
   //  Creates a Vector object to store the points.
   //-----------------------------------------------------------------
   public void init()
   {
      pointList = new Vector();
      count = 0;

      addMouseListener(this);

      setBackground (Color.black);
      setSize (APPLET_WIDTH, APPLET_HEIGHT);
   }

   //-----------------------------------------------------------------
   //  Draws all of the dots stored in the Vector.
   //-----------------------------------------------------------------
   public void paint (Graphics page)
   {
      page.setColor (Color.green);

      // Retrieve an iterator for the vector of points
      Iterator pointIterator = pointList.iterator();

      while (pointIterator.hasNext())
      {
         Point drawPoint = (Point) pointIterator.next();
         page.fillOval (drawPoint.x - RADIUS, drawPoint.y - RADIUS,
                        RADIUS * 2, RADIUS * 2);
      }

      page.drawString ("Count: " + count, 5, 15);
   }

   //-----------------------------------------------------------------
   //  Adds the current point to the list of points and redraws the
   //  applet whenever the mouse is pressed.
   //-----------------------------------------------------------------
   public void mousePressed (MouseEvent event)
   {
      pointList.addElement (event.getPoint());
      count++;
      repaint();
   }

   //-----------------------------------------------------------------
   //  Provide empty definitions for unused event methods.
   //-----------------------------------------------------------------
   public void mouseClicked (MouseEvent event) {}
   public void mouseReleased (MouseEvent event) {}
   public void mouseEntered (MouseEvent event) {}
   public void mouseExited (MouseEvent event) {}
}
```

```
<! Dots2.html>
<HTML>

<HEAD>
<TITLE>The Dots2 Applet</TITLE>
</HEAD>

<BODY>
<center>
<H3>The Dots2 Applet:</H3>
<APPLET CODE="Dots2.class" WIDTH=200 HEIGHT=100></APPLET>
<HR>
</center>
</BODY>

</HTML>
```