screen sizes are 17, 19, 20, and 24 inches. (One possible outcome to be displayed in the text box is "You have a Dell computer with 2 GB of memory and a 17-inch monitor.")

Solutions to Practice Problems 9.1

1. Yes, if all the items in the list box are distinct. However, if an item is repeated and the second occurrence is selected, then the first statement will delete that item, whereas the second statement will delete the earlier occurrence of the item.

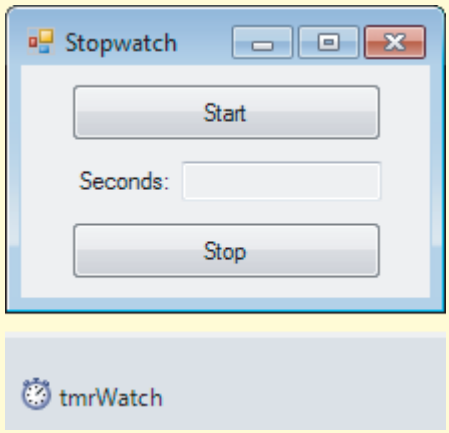
9.2 Eight Additional Controls and Objects

The Timer Control

VideoNote
Additional controls

The **timer** control, which is not visible on the form during run time, raises an event after a specified amount of time has passed. (The timer control is found only in the *All Windows Forms* group of the Toolbox. When you double-click on the timer control in the Toolbox, it appears in the **component tray**, at the bottom of the Form Designer.) The length of time, measured in milliseconds, is set with the Interval property to be any integer from 1 to 2,147,483,647 (about 596 hours). The event raised each time Timer1.Interval milliseconds elapses is called Timer1.Tick. In order to begin timing, a timer must first be turned on by setting its Enabled property to True. A timer is turned off by setting its Enabled property to False. The standard prefix for the name of a timer control is *tmr*.

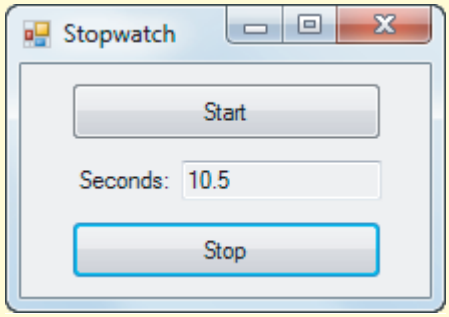
Example 1 The following program creates a stopwatch that updates the time every tenth of a second.



OBJECT	PROPERTY	SETTING
frmStopwatch	Text	Stopwatch
btnStart	Text	Start
lblSeconds	Text	Seconds:
txtSeconds	ReadOnly	True
btnStop	Text	Stop
tmrWatch	Interval	100

```
Private Sub btnStart_Click(...) Handles btnStart.Click
  txtSeconds.Text = "0"     'Reset watch
  tmrWatch.Enabled = True
End Sub

Private Sub btnStop_Click(...) Handles btnStop.Click
  tmrWatch.Enabled = False
End Sub

Private Sub tmrWatch_Tick(...) Handles tmrWatch.Tick
  'Next line displays the time rounded to one decimal place
  txtSeconds.Text = CStr((CDbl(txtSeconds.Text) + 0.1))
End Sub
```

[Run, click on the Start button, wait 10.5 seconds, and click on the Stop button.]



The Random Class

Visual Basic has a useful object called a random number generator that is declared with a statement of the form

```
Dim randomNum As New Random
```

If m and n are whole numbers, with $m < n$, then the value of

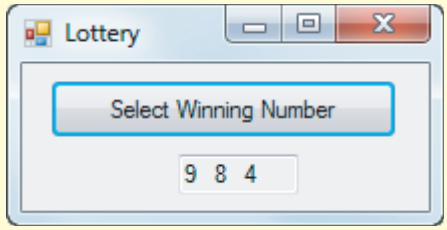
```
randomNum.Next(m, n)
```

is a randomly selected whole number from m through n, including m but excluding n. The Next method of this built-in object allows us to produce some interesting applications.

Example 2 A lottery number is obtained by selecting a Ping-Pong ball from each of three separate bowls. Each ball is numbered with an integer from 1 through 9. The following program produces a lottery number. Such a program is said to **simulate** the selection of Ping-Pong balls.

```
Private Sub btnSelect_Click(...) Handles btnSelect.Click
  'Display the winning lottery number
  Dim randomNum As New Random
  Dim num1, num2, num3 As Integer
  num1 = randomNum.Next(1, 10)
  num2 = randomNum.Next(1, 10)
  num3 = randomNum.Next(1, 10)
  txtNumbers.Text = num1 & "   " & num2 & "   " & num3
End Sub
```

[Run, and then click on the button.]



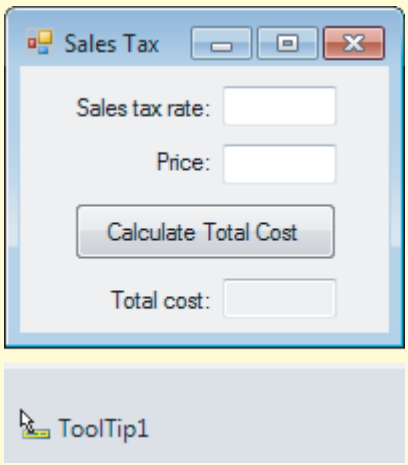
The ToolTip Control

The Visual Basic IDE uses tooltips to identify buttons on the Toolbar and icons in the Toolbox. When we hover the mouse over one of these items, a small rectangular box appears after $\frac{1}{2}$ second and remains visible for 5 seconds. The ToolTip control allows us to create tooltips for the controls in our programs.

A ToolTip Walkthrough

1. Start a new program. (There is no need to give a name to the program.)
2. Double-click on the ToolTip control in the Toolbox. (The control will appear with the default name ToolTip1 in the component tray at the bottom of the Form Designer.)
3. Place controls on the form and enter code as shown in Example 3. (The setting for the "ToolTip on ToolTip1" property of txtRate holds the information that will appear when the mouse hovers over the text box.)

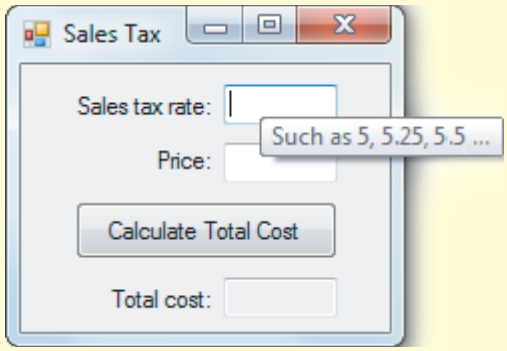
Example 3 In the following program, suppose the sales tax rate is 5%. The user might not know whether to enter 5 or .05 into the text box. A tooltip comes to the rescue.



OBJECT	PROPERTY	SETTING
frmName	Text	Sales Tax
lblRate	Text	Sales tax rate:
txtRate	ToolTip on Tooltip1	Such as 5, 5.25, 5.5 ....
lblPrice	Text	Price:
txtPrice		
btnCalculate	Text	Calculate Total Cost
lblTotalCost	Text	Total cost:
txtTotalCost	ReadOnly	True

```
Private Sub btnCalculate_Click(...) Handles btnCalculate.Click
  Dim taxRate As Double = CDbl(txtRate.Text) / 100
  Dim price As Double = CDbl(txtPrice.Text)
  Dim totalCost As Double = price + taxRate * price
  txtTotalCost.Text = FormatCurrency(totalCost)
End Sub
```

[Run, and hover the mouse over the first text box.]



Normally, a control placed on a form does not have a "ToolTip on ToolTip1" property. That property appears in the Properties window only after a ToolTip control has been added to the component tray.

In the example above, we created a tooltip for only one of the text boxes. However, we could have specified a tooltip for the other controls.

By default, a tooltip appears $^1/_2$ second after the cursor hovers over a control and lasts for 5 seconds. These two durations can be altered with the Tooltip control's AutomaticDelay and AutoPopDelay properties. The numeric settings for these two properties are in milliseconds. Specifically, the AutomaticDelay setting determines the length of time required for a tooltip to appear, and the AutoPopDelay setting determines the length of time the tooltip remains visible while the cursor is stationary inside a control.

The Clipboard

The **Clipboard** object is used to copy or move information from one location to another. It is maintained by Windows and therefore even can be used to transfer information from one Windows application to another. It is actually a portion of memory that holds information and has no properties or events.

If *str* is a string, then the statement

```
Clipboard.SetText(str)
```

replaces any text currently in the Clipboard with the value of *str*. The statement

```
str = Clipboard.GetText
```

assigns the text in the Clipboard to the string variable *str*. The statement

```
Clipboard.SetText("")
```

deletes the contents of the Clipboard.

A portion of the text in a text box or combo box can be **selected** by dragging the mouse across it or by moving the cursor across it while holding down the Shift key. After you select text, you can place it into the Clipboard by pressing Ctrl + C. Also, if the cursor is in a text box and you press Ctrl + V, the contents of the Clipboard will be pasted at the cursor position. These tasks also can be carried out in code. The SelectedText property of a text box holds the selected string from the text box, and a statement such as

```
Clipboard.SetText(txtBox.SelectedText)
```

copies this selected string into the Clipboard. The statement

```
txtBox.SelectedText = Clipboard.GetText
```

replaces the selected portion of txtBox with the contents of the Clipboard. If nothing has been selected, the statement inserts the contents of the Clipboard into txtBox at the cursor position.

The Picture Box Control

The **picture box** control is designed to hold drawings created with graphics commands or pictures stored in graphics files such as bmp files created with Windows Paint, ico files of icons that come with Windows, or gif and jpeg images used on the World Wide Web. By convention, names of picture box controls have the prefix *pic*.

A statement of the form

```
picBox.CreateGraphics.DrawRectangle(Pens.Blue, x, y, w, h)
```

where *x*, *y*, *w*, *h* are of type Integer, draws a blue rectangle of width *w* pixels and height *h* pixels in the picture box. The upper-left corner of the rectangle will be *x* pixels from the left side and *y* pixels from the top side of the picture box. (See Fig. 9.7.) To get a feel for how big a pixel is, the initial size of the form when you create a new project is 300 pixels by 300 pixels.

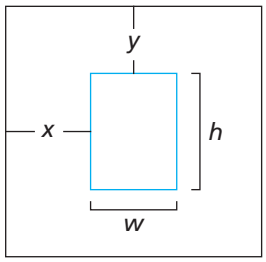


FIGURE 9.7 **Rectangle arguments.**

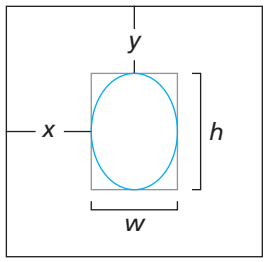


FIGURE 9.8 **Ellipse arguments.**

The color Blue can be replaced by other colors. If DrawRectangle is replaced with DrawEllipse, the statement above will draw the ellipse that would be inscribed in the rectangle just described. See Fig. 9.8. Also, if *w* and *h* have the same value, the ellipse will be a circle with a diameter of that value. In Fig. 9.9, the picture box has a size of 140 by 140 pixels, a white background, and a FixedSingle border style. The circle is drawn with the statement

```
picBox.CreateGraphics.DrawEllipse(Pens.Red, 35, 35, 70, 70)
```

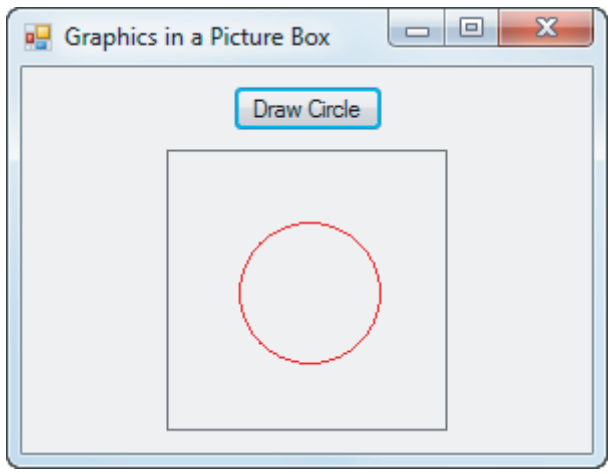


FIGURE 9.9 **Picture box containing red circle.**

A picture can be placed in a picture box control with the Image property. If you double-click on the Image property during design time, an Open dialog box appears and assists you in selecting an appropriate file. However, prior to setting the Image property, you should set the SizeMode property. If the SizeMode property is set to AutoSize, the picture box control will be resized to fit the picture. If the SizeMode property is set to StretchImage, the picture will be resized to fit the picture box control. Therefore, with the StretchImage setting, pictures can be reduced (by placing them into a small picture box control) or enlarged (by placing them into a picture box control bigger than the picture). Figure 9.10 shows a picture created with Paint and reduced by StretchImage to fit the picture box.

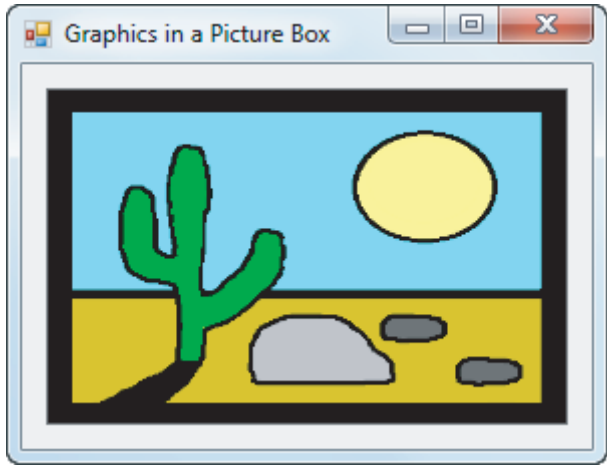


FIGURE 9.10 **Picture box with desert scene.**

A picture also can be assigned to a picture box control at run time. However, a statement such as

```
picBox.Image = filespec
```

will not do the job. Instead, we must create an Image object with a statement such as

```
picBox.Image = Image.FromFile(filespec)
```

The SizeMode property can be altered at run time with a statement such as

```
picBox.SizeMode = PictureBoxSizeMode.AutoSize
```

The MenuStrip Control

Visual Basic forms can have menu bars similar to those in most Windows applications. Figure 9.11 shows a typical menu bar, with the *Order* menu revealed. Here, the menu bar contains two menu items (*Order* and *Color*), referred to as **top-level** menu items. When the *Order* menu item is clicked, a drop-down list containing two second-level menu items (*Ascending* and *Descending*) appears. Although not visible here, the drop-down list under *Color* contains the two second-level menu items *Foreground* and *Background*. Each menu item is treated as a distinct control that responds to a Click event. The Click event is raised not only by the click of the mouse button, but also for top-level items by pressing Alt + *accessKey* and for second-level items by just pressing the access key. The event procedure for the *Ascending* or *Descending* menu item also can be raised by pressing the shortcut key combination Ctrl + A or Ctrl + D.

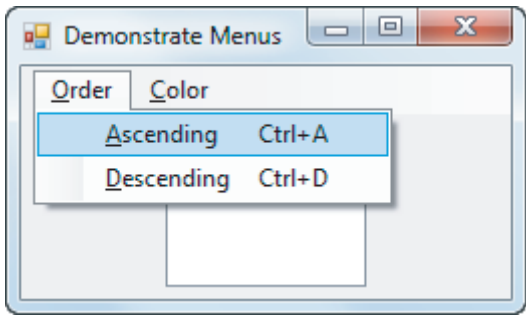


FIGURE 9.11 **A simple menu.**

Menus are created with the MenuStrip control—the third control in the *Menus & Toolbars* group of the Toolbox. Each menu item has a Text property (what the user sees) and a Name property (used to refer to the item in code.) The following walkthrough creates the menu bar in Fig. 9.11:

1. Start a new program.
2. Double-click on the MenuStrip control in the Toolbox. The control appears in the component tray, and a menu designer appears just below the title bar in the Form Designer. See Fig. 9.12.

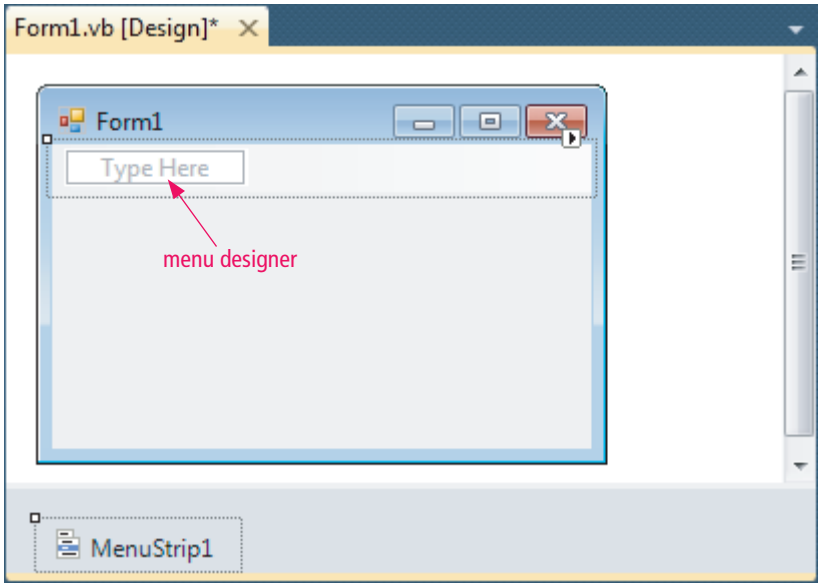


FIGURE 9.12 **The MenuStrip control added to a form.**

3. Click on the rectangle that says "Type Here", type in "&Order", and press the Enter key. (The ampersand specifies O as an access key for the menu item.) "Type Here" rectangles appear below and to the right of the *Order* menu item. The rectangle below the *Order* menu item is used to create a second-level item for the *Order* menu. The rectangle on the right is used to create a new first-level menu item.
4. Type "&Ascending" into the rectangle below the *Order* rectangle, and press the Enter key.
5. Click on the *Ascending* menu item to display its Property window. In the Property window, change the Name property of the menu item from AscendingToolStripMenuItem to mnuOrderAsc. Also, click on the down-arrow at the right of the ShortcutKeys Settings box, click on the *Ctrl* check box under Modifiers, click on the down-arrow button of the Key combo box, click on *A* in the drop-down list, and press the Enter key. ("Ctrl + A" will appear to the right of the word *Ascending*.)
6. Type "&Descending" into the rectangle below the *Ascending* rectangle, set the Name property of the *Descending* menu item to mnuOrderDesc, and set the ShortcutKeys Property to Ctrl + D.
7. Click on the rectangle to the right of the *Order* rectangle and enter the text "&Color".
8. Type "&Foreground" into the rectangle below the *Color* rectangle, and set its Name property to mnuColorFore.
9. Type "&Background" into the rectangle below the *Foreground* rectangle, and set its Name property to mnuColorBack.

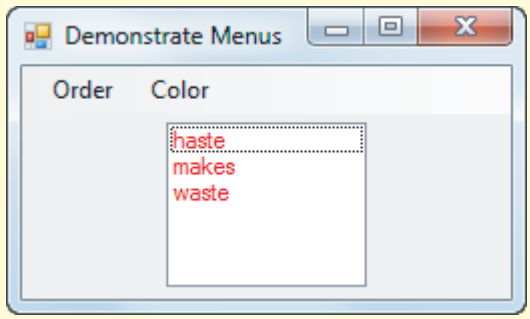
10. Click on the *Foreground* rectangle, and type "&Red" into the rectangle on its right. (We have just created a third-level menu item.) Set its Name property to mnuColorForeRed.
11. Type "&Blue" into the rectangle below the *Red* rectangle, and set its Name property to mnuColorForeBlue.
12. Click on the *Background* rectangle, type "&Yellow" into the rectangle on its right, and set its Name property to mnuColorBackYellow.
13. Type "&White" into the rectangle below the *Yellow* rectangle, and set its Name property to mnuColorBackWhite. Then set its Checked property to True. A check mark will appear to the left of the word "White".
14. Run the program; click on *Order* to see its menu items; click on *Color* and hover over the word *Foreground* to see its menu items. The menu items are useful only after we write code for their Click event procedures.

Example 4 The following program uses the menu just created to alter the color of the text in a list box and the order of its items. The form has the text "Demonstrate Menus" in its title bar.

```
Private Sub frmDemo_Load(...) Handles MyBase.Load
  lstOutput.Items.Add("makes")
  lstOutput.Items.Add("haste")
  lstOutput.Items.Add("waste")
End Sub

Private Sub mnuOrderAsc_Click(...) Handles mnuOrderAsc.Click
  lstOutput.Sorted = True
End Sub

Private Sub mnuOrderDesc_Click(...) Handles mnuOrderDesc.Click
  'This code uses the fact that if a list is in ascending order,
  'then displaying it backwards gives a descending list
  Dim temp(2) As String     'Hold ascending array of items
  lstOutput.Sorted = True   'Sort the items alphabetically
  For i As Integer = 0 To 2 'Place sorted items into the array
    temp(i) = CStr(lstOutput.Items(i))
  Next
  lstOutput.Sorted = False  'Turn off the Sorted property
  lstOutput.Items.Clear()
  For i As Integer = 2 To 0 Step -1
    lstOutput.Items.Add(temp(i))
  Next
End Sub

Private Sub mnuColorForeRed_Click(...) Handles mnuColorForeRed.Click
  lstOutput.ForeColor = Color.Red
End Sub

Private Sub mnuColorForeBlue_Click(...) Handles mnuColorForeBlue.Click
  lstOutput.ForeColor = Color.Blue
End Sub

Private Sub mnuColorBackYellow_Click(...) Handles _
            mnuColorBackYellow.Click
  'Make Yellow the background color of the list box, guarantee that a
```

```
  'check mark appears in front of the menu item Yellow and not in front
  'of the White menu item
  lstOutput.BackColor = Color.Yellow
  mnuColorBackYellow.Checked = True
  mnuColorBackWhite.Checked = False
End Sub

Private Sub mnuColorBackWhite_Click(...) Handles mnuColorBackWhite.Click
  lstOutput.BackColor = Color.White
  mnuColorBackYellow.Checked = False
  mnuColorBackWhite.Checked = True
End Sub
```

[Run, click on *Ascending* in the *Order* menu, click on the *Color* menu, hover over *Foreground*, and click on *Red*.]



The Horizontal and Vertical Scroll Bar Controls

Figure 9.13 shows the two types of **scroll bars**. When the user clicks on one of the arrow buttons, the scroll box moves a small distance toward that arrow. When the user clicks between the scroll box and one of the arrow buttons, the scroll box moves a large distance toward that

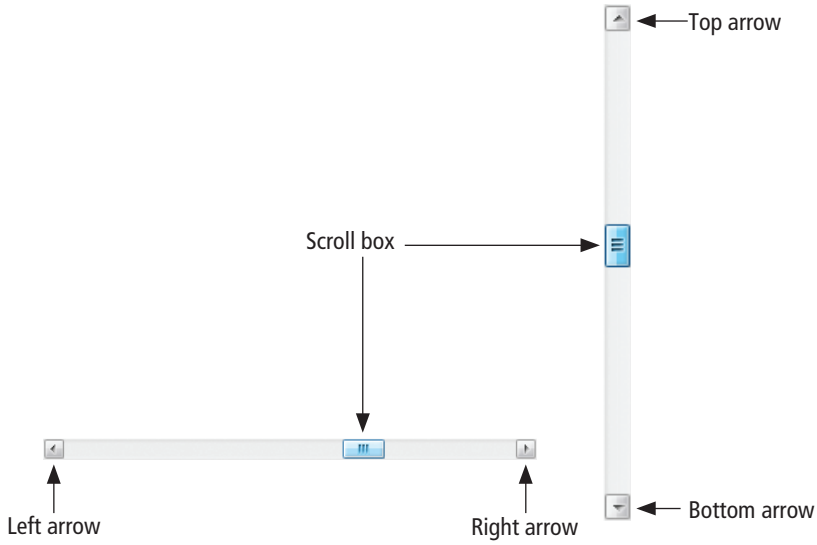


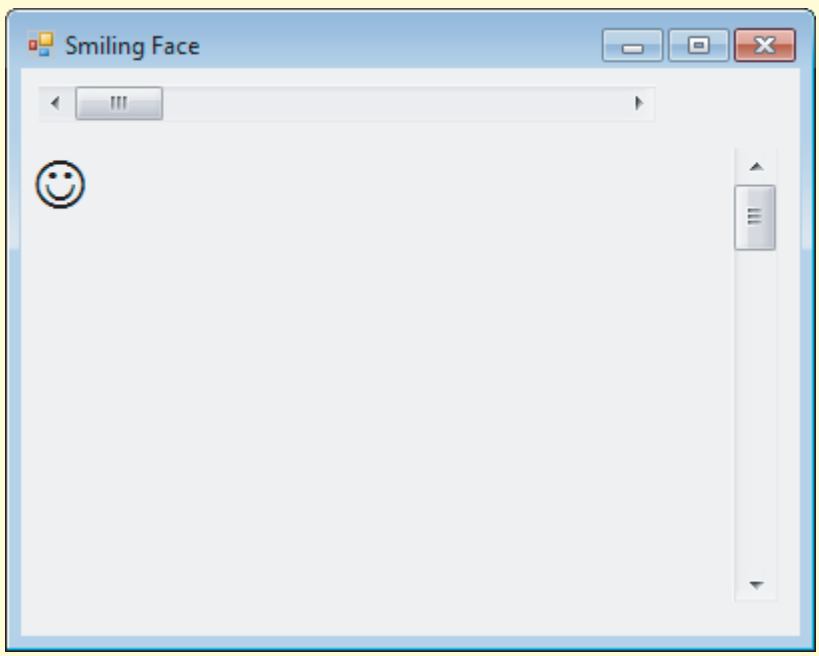
FIGURE 9.13 Horizontal and vertical scroll bars.

arrow. The user can also move the scroll box by dragging it. The main properties of a scroll bar control are Minimum, Maximum, Value, SmallChange, and LargeChange, which are set to integers. The standard prefix for the name of a scroll bar is *hsb* or *vsb*. At any time, hsbBar.Value is a number between hsbBar.Minimum and hsbBar.Maximum determined by the position of the left side of the scroll box. If the left side of the scroll box is halfway between the two arrows, then hsbBar.Value is a number halfway between hsbBar.Minimum and hsbBar.Maximum. If the scroll box is near the left arrow button, then hsbBar.Value is an appropriately proportioned value near hsbBar.Minimum. When the user clicks on an arrow button, hsbBar.Value changes by hsbBar.SmallChange and the scroll box moves accordingly. When the bar between the scroll box and one of the arrows is clicked, hsbBar.Value changes by hsbBar.LargeChange and the scroll box moves accordingly. When the scroll box is dragged, hsbBar.Value changes accordingly. The default values of Minimum, Maximum, Value, SmallChange, and LargeChange are 0, 100, 0, 1, and 10, respectively. However, these values are usually changed at design time. The width of the scroll box is equal to the value of LargeChange. Since hsbBar.Value is determined by the left side of the scroll box, the greatest value it can assume is (hsbBar.Maximum – hsbBar.LargeChange + 1). Vertical scroll bars behave similarly.

Note: The setting for the Minimum property must be less than the setting for the Maximum property. The Minimum property determines the values for the left and top arrows. The Maximum property determines the values for the right and bottom arrows.

The two controls are referred to as HScrollBar and VScrollBar in the Toolbox. Their default event, Scroll, is raised whenever the user clicks on any part of the scroll bar.

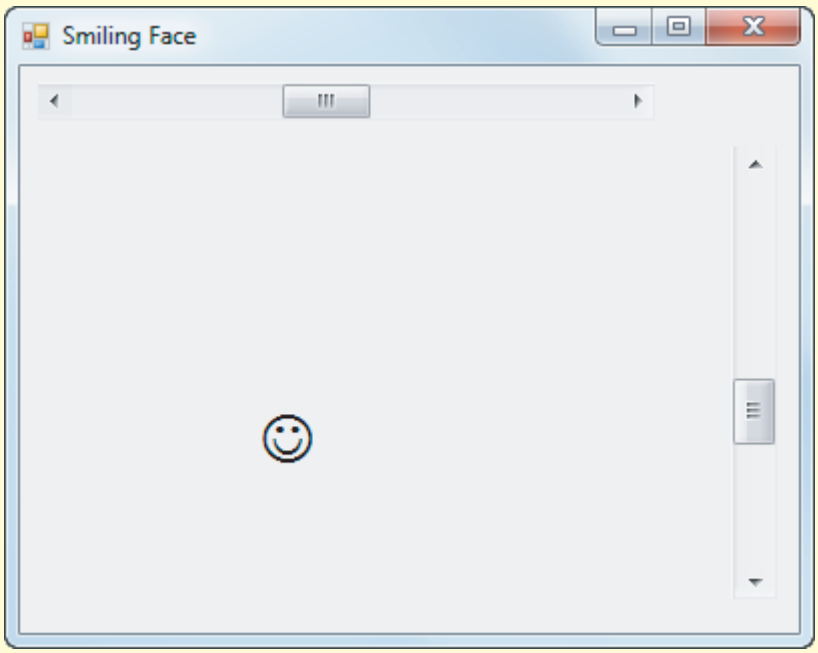
Example 5 The following program uses scroll bars to move a smiling face around the form. The face is a large Wingdings character J inside a label. The values lblFace.Left and lblFace.Top are the distances in pixels of the label from the left side and top of the form.



OBJECT	PROPERTY	SETTING
frmFace	Text	Smiling Face
hsbXPos	Minimum	0
	Maximum	300
	SmallChange	10
	LargeChange	50
	Value	0
vsbYPos	Minimum	30
	Maximum	300
	SmallChange	10
	LargeChange	50
	Value	30
lblFace	Text	J
	Font	Wingdings, 24pt

```
Private Sub hsbXpos_Scroll(...) Handles hsbXpos.Scroll
  lblFace.Left = hsbXpos.Value
End Sub

Private Sub vsbYpos_Scroll(...) Handles vsbYpos.Scroll
  lblFace.Top = vsbYpos.Value
End Sub
```

[Run, and move the scroll boxes on the scroll bars.]



Practice Problems 9.2

1. What is the effect of the following event procedure?

```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim randomNum As New Random
  Dim contestant() As String = {"Mary", "Pat", "Linda",
                                "Barbara", "Maria"}
```

```
    Dim number As Integer, temp As String
    For i As Integer = 0 To 3
      number = randomNum.Next(i, 5)
      temp = contestant(i)
      contestant(i) = contestant(number)
      contestant(number) = temp
    Next
    lstOutput.Items.Clear()
    For i As Integer = 0 To 4
      lstOutput.Items.Add(contestant(i))
    Next
End Sub
```

EXERCISES 9.2

In Exercises 1 through 6, determine the effect of setting the property to the value shown.

1. Timer1.Interval = 5000
2. Timer1.Enabled = False
3. ToolTip1.AutomaticDelay = 1000
4. ToolTip1.AutoPopDelay = 4000
5. mnuOrderAsc.Checked = True
6. mnuOrderAsc.Checked = False

In Exercises 7 through 25, describe the effect of executing the statement(s).

7.
```
Timer1.Interval = CInt(intVar * 1000)
```

8.
```
Dim randomNum As New Random
txtBox.Text = CStr(randomNum.Next(1, 101))
```

9.
```
Dim randomNum As New Random
Dim number As Integer
'Assume the array pres() contains the names of the 44 U.S. presidents
number = randomNum.Next(0, 44)
txtBox.Text = pres(number)
```

10.
```
Dim randomNum As New Random
'95 characters can be produced by the computer keyboard
txtBox.Text = Chr(randomNum.Next(32, 127))
```

11.
```
Dim randomNum As New Random
Dim number As Integer, temp As String
'Suppose the array states() contains the names of the 50 states
number = randomNum.Next(0, 50)
lstBox.Items.Add(states(number))
temp = states(number)
states(number) = states(49)
states(49) = temp
lstBox.Items.Add(states(randomNum.Next(0, 49)))
```

12.
```
Dim randomNum As New Random
Dim suit() As String = {"Hearts", "Clubs", "Diamonds", "Spades"}
Dim denomination() As String = {"2", "3", "4", "5", "6",
     "7", "8", "9", "10", "Jack", "Queen", "King", "Ace"}
txtBox.Text = denomination(randomNum.Next(0, 13)) & " of " &
              suit(randomNum.Next(0, 4))
```

13.
```
Clipboard.SetText("")
```

14.
```
Clipboard.SetText("Hello")
```

15.
```
Clipboard.SetText(txtBox.SelectedText)
```

16.
```
txtBox.SelectedText = Clipboard.GetText()
```

17.
```
txtBox.Text = Clipboard.GetText
```

18.
```
Dim strVar As String = "Happy"
Clipboard.SetText(strVar)
```

19.
```
Dim strVar As String
strVar = Clipboard.GetText
```

20.
```
PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
```

21.
```
PictureBox1.CreateGraphics.DrawEllipse(Pens.Blue, 20, 30, 100, 100)
```

22.
```
PictureBox1.CreateGraphics.DrawRectangle(Pens.Green, 25, 50, 200, 100)
```

23.
```
PictureBox1.Image = Image.FromFile("Airplane.bmp")
```

24.
```
HScrollBar2.Value = CInt((HScrollBar2.Maximum + HScrollBar2.Minimum) / 2)
```

25.
```
VScrollBar2.SmallChange = VScrollBar2.LargeChange
```

In Exercises 26 through 44, write one or more lines of code to carry out the task.

26. Specify that Timer1 raise an event every half second.
27. Specify that Timer1 cease to raise the Tick event.
28. The array *names* contain twenty names. Display a randomly selected name in txtBox.
29. Display in txtBox a randomly selected number from 1 to 12.
30. Display in txtBox a letter randomly selected from the alphabet.
31. The file Towns.txt contains the names of twenty-five cities. Display a randomly selected city in txtBox.
32. Display in txtBox the sum of the faces after tossing a pair of dice.
33. Suppose the array *rivers* contains the names of rivers. Randomly select two different rivers from the array and display them in lstBox.
34. Replace the selected portion of txtBox with the contents of the Clipboard.
35. Clear the contents of the Clipboard.
36. Place the word "Rosebud" into the Clipboard.
37. Copy the selected text in txtBox into the Clipboard.
38. Delete the selected portion of txtBox.
39. Assign the contents of the Clipboard to the integer variable *amount*.
40. Draw a yellow circle of diameter 100 pixels in PictureBox1.
41. Remove the check mark in front of the menu item named mnuOrderDesc.
42. Change the text for mnuOrderDesc to "Decreasing Order".
43. Move the scroll box of VScrollBar2 as high as possible.
44. Move the scroll box of HScrollBar2 one-third of the way from the left arrow to the right arrow.

Exercises 45 and 46 refer to Example 4.

45. Make a conjecture on the effect of the following statement and test your conjecture.

```
mnuOrderAsc.Enabled = False
```

46. Make a conjecture on the effect of the following statement and test your conjecture.

```
mnuOrderAsc.Visible = False
```

47. Write a program to create a decorative digital clock. The clock in the Digital Clock form in Fig. 9.14 is inserted in a picture box control containing the Trees.bmp picture. The values for hour, minute, and second can be obtained as Hour(Now), Minute(Now), and Second(Now).

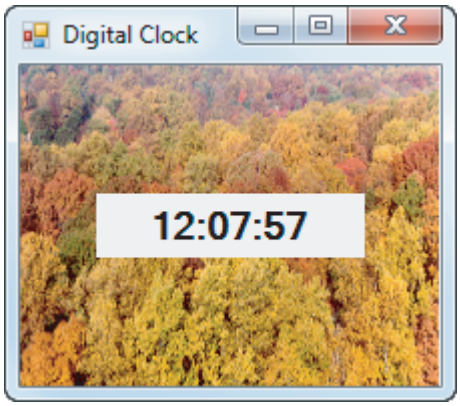


FIGURE 9.14 **Sample run of Exercise 47.**

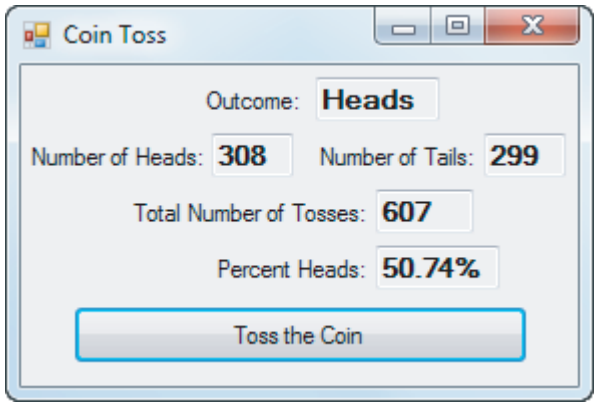


FIGURE 9.15 **Sample run of Exercise 48.**

48. Write a program using the form in Fig. 9.15. Each time the button is pressed, a Random object is used to simulate a coin toss and the values are updated. The figure shows the status after the button has been pressed 607 times. ***Note:*** You can produce tosses quickly by just holding down the Enter key. Although the percentage of heads initially will fluctuate considerably, it should stay close to 50% after many (say, 1000) tosses.

49. The file Members.txt contains the names of the members of a large club. Write a program to randomly select people to serve as President, Treasurer, and Secretary. ***Note:*** A person cannot hold more than one office.

50. Place the names of the 52 playing cards into the array *deckOfCards*. Then display the names of five randomly chosen cards in lstPokerHand.

51. Write a program that repeatedly rolls a pair of dice and tallies the number of rolls and the number of those rolls that total seven. The program should stop when 1000 sevens have been rolled, and then report the approximate odds of rolling a seven. (The approximate odds will be "1 in " followed by the result of dividing the number of rolls by the number of rolls that came up seven.)

52. Write a program to randomly select 40 different people from a group of 100 people whose names are contained in the text file Names.txt.

53. *The Birthday Problem.* Given a random group of 23 people, how likely is it that two people have the same birthday? To answer this question, write a program that creates an array of 23 elements, randomly assigns to each subscripted variable one of the integers from 1 through 365, and checks to see if any of the subscripted variables have the same value. (Make the simplifying assumption that no birthdays occur on February 29.) Now expand the program to repeat the process 1000 times and determine the percentage of the time that there is a match.

VideoNote
Blackjack (Homework)

54. Consider a carnival game in which two cards are drawn at random from a deck of 52 cards. If either one or both of the cards is a diamond, you win one dollar. If neither card is a diamond, you lose one dollar. Simulate playing the game 1000 times and determine how much money you win or lose.

55. Write a program containing text boxes named txtName and txtZipCode. The tooltip "Enter your full name." or "Enter your 9-digit zip code." should appear when the mouse hovers over the corresponding text box.

56. Write a program containing a list box. The tooltip "Double-click on an item to delete it from the list." should appear when the mouse hovers over the list box.

57. Write a program to display a picture (contained in a .bmp file on the hard drive) in a picture box. The .bmp file should be selected with an OpenFileDialog control.

58. Write a program with a single text box and a menu having the single top-level item *Edit* and the three second-level items *Copy*, *Paste*, and *Cut*. *Copy* should place a copy of the selected portion of the text box into the Clipboard, *Paste* should duplicate the contents of the Clipboard at the cursor position, and *Cut* should delete a selected portion of the text box and place it in the Clipboard.

59. The Ch09\Pictures folder contains files named Moon1.bmp, Moon2.bmp, . . . , Moon8.bmp, which show eight phases of the moon. Create a form consisting of a picture box control and a timer control. Every two seconds assign another file to the Image property of the picture box control to see the moon cycle through its phases every 16 seconds. One phase is shown in Fig. 9.16.

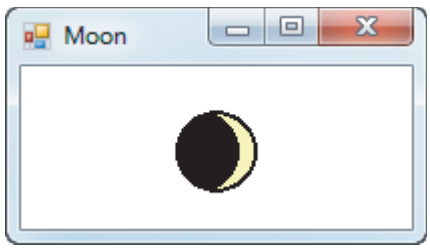


FIGURE 9.16 **Sample run of Exercise 59.**

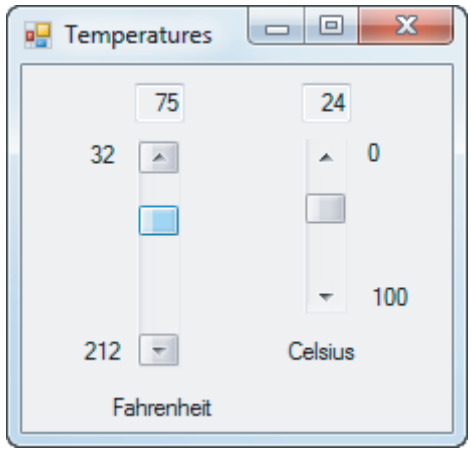


FIGURE 9.17 **Sample run of Exercise 60.**

60. Write a program to synchronize the two thermometers shown in the Temperatures form in Fig. 9.17. When the scroll box of either thermometer is moved, the other thermometer should move to the corresponding temperature, and the two temperatures displayed above the thermometers should be updated. ***Note***: F = (9/5)C + 32.

61. *Simulation of the Times Square Ball*. Create a form with a vertical scroll bar and a timer control. When the program is run, the scroll box should be at the top of the scroll bar. Each second the scroll box should descend one-tenth of the way down. When the scroll box reaches the bottom after 10 seconds, a message box displaying HAPPY NEW YEAR should appear.

Solution to Practice Problem 9.2

1. The event procedure places the names of the contestants in a list box in a random order.