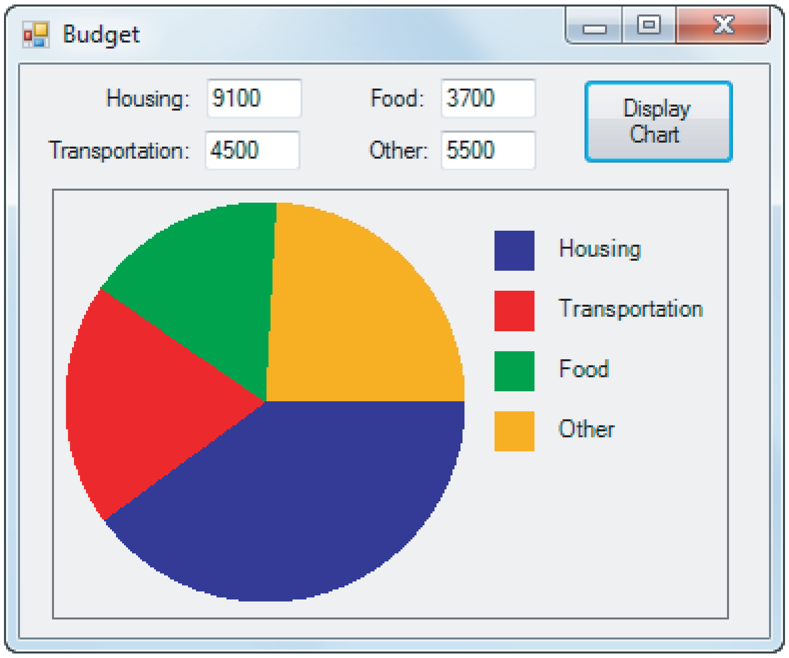


FIGURE 9.47 Form for Exercise 18.

Solutions to Practice Problems 9.4

1. False. Only the Fill methods and the DrawString method use colored Brushes. The DrawLine method uses colored Pens.
2.
```
Dim gr As Graphics = Me.CreateGraphics
Dim fnt As Font = New Font("Wingdings", 20)
gr.DrawString("J", fnt, Brushes.Red, 0, 0)
```

CHAPTER 9 SUMMARY

1. *List boxes* provide easy access to lists of data. Items() holds the items stored in the list box. Each item is identified by an index number. The lists can be automatically sorted (Sorted property = True) and altered (Items.AddItem, Items.RemoveAt, and Items.Remove methods), the currently highlighted item identified (Text property), and the number of items determined (Items.Count property).
2. Simple and DropDown style *combo boxes* are enhanced text boxes. They allow the user to fill the text box by selecting an item from a list or by typing the item directly into the text box. The contents of the text box are assigned to the combo box's Text property. A DropDownList style combo box is essentially a list box that drops down instead of being permanently displayed.
3. The *timer control* raises an event repeatedly after a specified time interval.
4. An object of type *Random* can generate a randomly selected integer from a specified range.
5. The *ToolTip control* allows the program to display guidance when the mouse hovers over a control.
6. The *Clipboard* is filled with the SetText method or by pressing Ctrl + C, and its contents are copied with the GetText method or by pressing Ctrl + V.

7. The *picture box control*, which displays pictures or geometric shapes, can expand to accommodate the size of a picture or have a picture alter its size to fit the control.

8. Menus, similar to the menus of the Visual Basic IDE itself, can be created with the *MenuStrip control*.

9. *Horizontal* and *vertical scroll bar controls* permit the user to select from among a range of integers by clicking or dragging with the mouse. The range is determined by the Minimum and Maximum properties. The Scroll event is raised by clicking on the scroll bar.

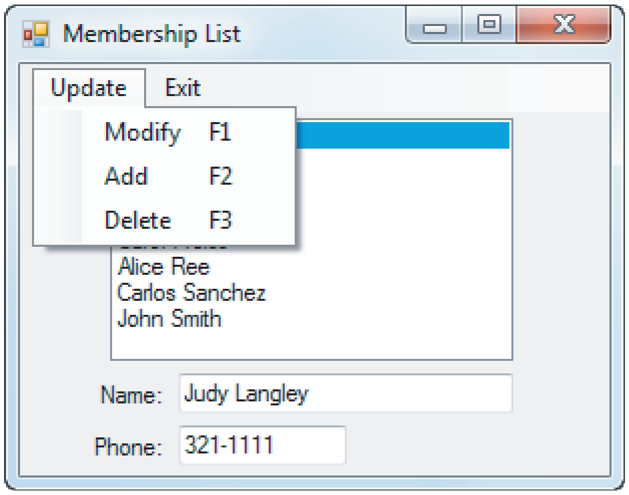
10. Additional forms can be added to a program to serve as customized dialog boxes. They are revealed with the *ShowDialog* method and removed with the *Close* method.

11. After a graphics object is produced with a *CreateGraphics* method, the *DrawString*, *DrawLine*, *FillRectangle*, *FillEllipse*, and *FillPie* methods can be used to display strings, lines, solid rectangles, solid ellipses, and solid sectors with colors supplied by Pen and Brush objects.

12. *Animation* can be produced by steadily moving a picture box containing an image.

CHAPTER 9 PROGRAMMING PROJECTS

1. *Membership List*. Write a menu-driven program to manage a membership list. See Fig. 9.48. Assume that the names and phone numbers of all members are stored in alphabetical order (by last name, then by first name) in the text file MemberPhones.txt. Each record consists of two fields—a name field and a phone number field. The names should appear in a list box when the form is loaded. When a name is highlighted, both the name and phone number of the person should appear in the text boxes at the bottom of the form. To delete a person, highlight his or her name and click on the *Delete* menu item. To change either a person's name or phone number, make the corrections in the text boxes and click on the menu item *Modify*. To add a new member, type his or her name and phone number into the text boxes and click on the menu item *Add*. When the *Exit* menu item is clicked, the new membership list should be written to the file and the program should terminate.



Alice Ree
Carlos Sanchez
John Smith

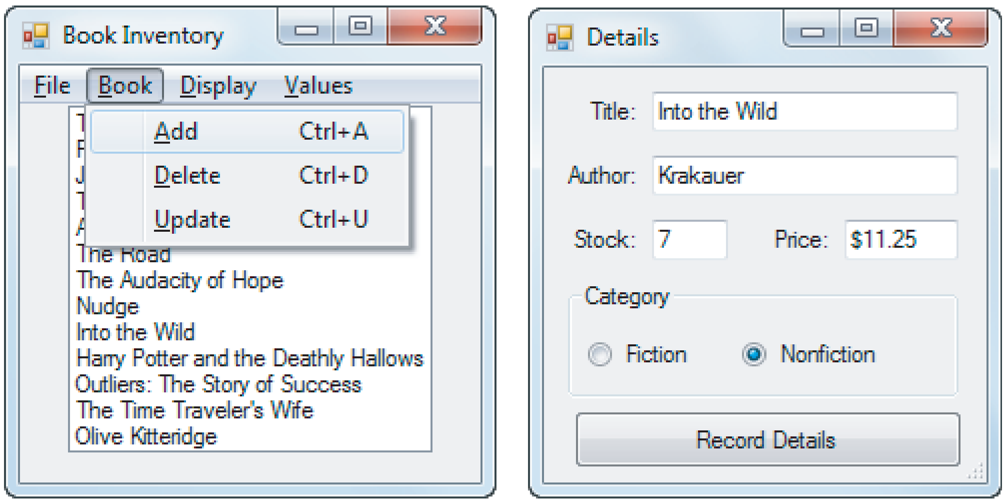
Name: Judy Langley

Phone: 321-1111

FIGURE 9.48 **Form for Programming Project 1.**

2. *Inventory Control*. Write a menu-driven multiform inventory program for a bookstore with data saved in a text file. Each record of the text file should consist of five fields—*title*, *author*, *category*, *wholesale price*, and *number in stock*. (The two categories are *fiction* and

nonfiction.) At any time, the program should display the titles of the books in stock in a list box. The user should have the option of displaying either all titles or just those in one of the two categories. The user should be able to add a new book, delete a book, or alter any of the fields of a book in stock. The adding and editing processes use the second form, frmDetails. See Fig. 9.49. At any time, the user should be able to calculate the total value of all books, or the total value of the books in either category. The menu item *File* contains the two second-level menu items *Save* and *Exit*. The menu items *Display* and *Values* each contain the three second-level menu items *All*, *Fiction*, and *Nonfiction*. (***Hint:*** Store the data about the books in an array of structures.)



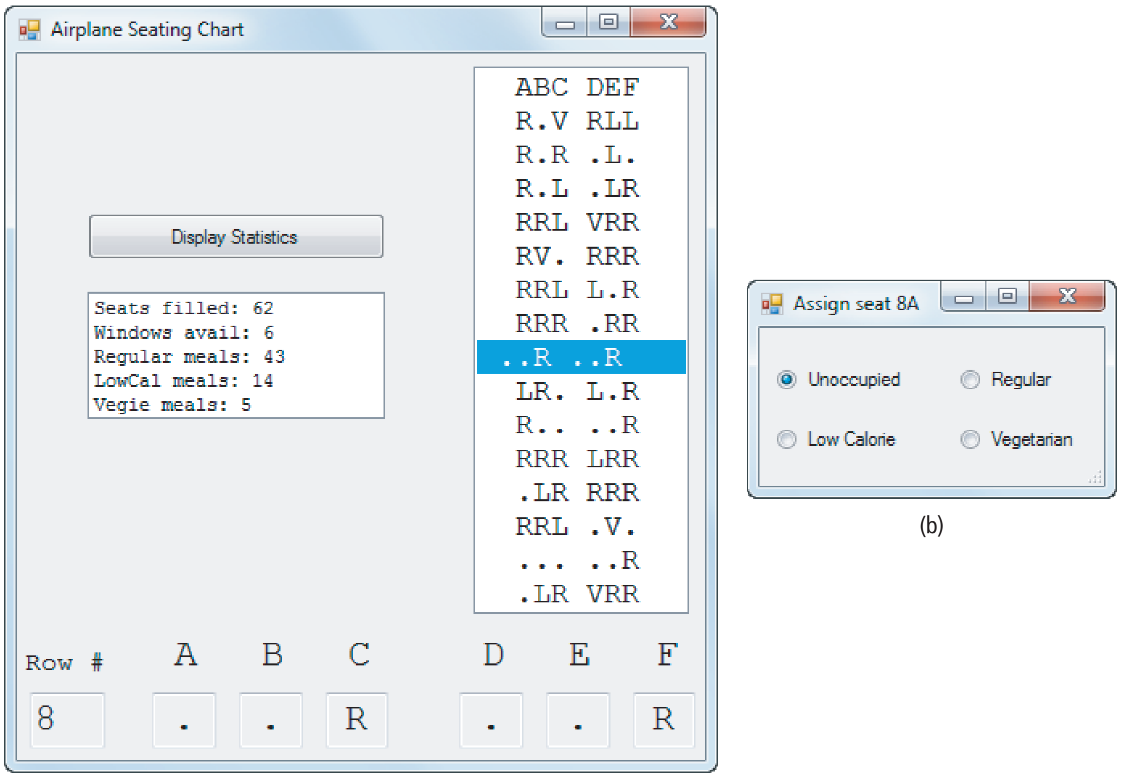
The Road
The Audacity of Hope
Nudge
Into the Wild
Harry Potter and the Deathly Hallows
Outliers: The Story of Success
The Time Traveler's Wife
Olive Kitteridge

Title: Into the Wild
Author: Krakauer
Stock: 7 Price: $11.25
Category
Fiction [✓ Nonfiction]

FIGURE 9.49 The two forms for Programming Project 2.

3. *Voting Machine*. The members of a club bring a computer to their annual meeting to use in the election of a new president. Write a program to handle the election. The program should add each candidate to a list box as he or she is nominated. After the nomination process is complete, club members should be able to approach the computer one at a time and double-click on the candidate of their choice. When a *Tally Votes* button is clicked on, a second list box, showing the number of votes received by each candidate, should appear alongside the first list box. Also, the name(s) of the candidate(s) with the highest number of votes should be displayed in a list box.

4. *Airplane Seating Chart*. An airplane has 15 rows (numbered 1 through 15), with six seats (labeled *A*, *B*, *C*, *D*, *E*, and *F*) in each row. Write a multiform program that keeps track of the seats that have been reserved and the type of meal requested by each passenger. The seating chart should be displayed in a list box with a line for each row. See Fig. 9.50(a). When the ticket agent clicks on the desired row in the list box, the row number and the status of the seats in the row should be displayed in seven read-only text boxes at the bottom of the form. When the agent clicks on one of the text boxes, a second form containing four option buttons labeled *Unoccupied*, *Regular*, *Low Calorie*, and *Vegetarian* should appear. See Fig. 9.50(b). Clicking on a radio button should close the second form and update both the text box and the row for that seat in the list box. Unoccupied seats are denoted with a period, and occupied seats are denoted with the first letter of their meal type. At any time, the agent should be able to request the number of seats filled, the number of window seats vacant, and the numbers of each type of meal ordered.



```
ABC DEF
R.V RLL
R.R .L.
R.L .LR
RRL VRR
RV. RRR
RRL L.R
RRR .RR
..R ..R
LR. L.R
R.. ..R
RRR LRR
.LR RRR
RRL .V.
... ..R
.LR VRR
```

```
Seats filled: 62
Windows avail: 6
Regular meals: 43
LowCal meals: 14
Vegie meals: 5
```

Row #	A	B	C	D	E	F
8	.	.	R	.	.	R

[✓ Unoccupied] Regular Low Calorie Vegetarian

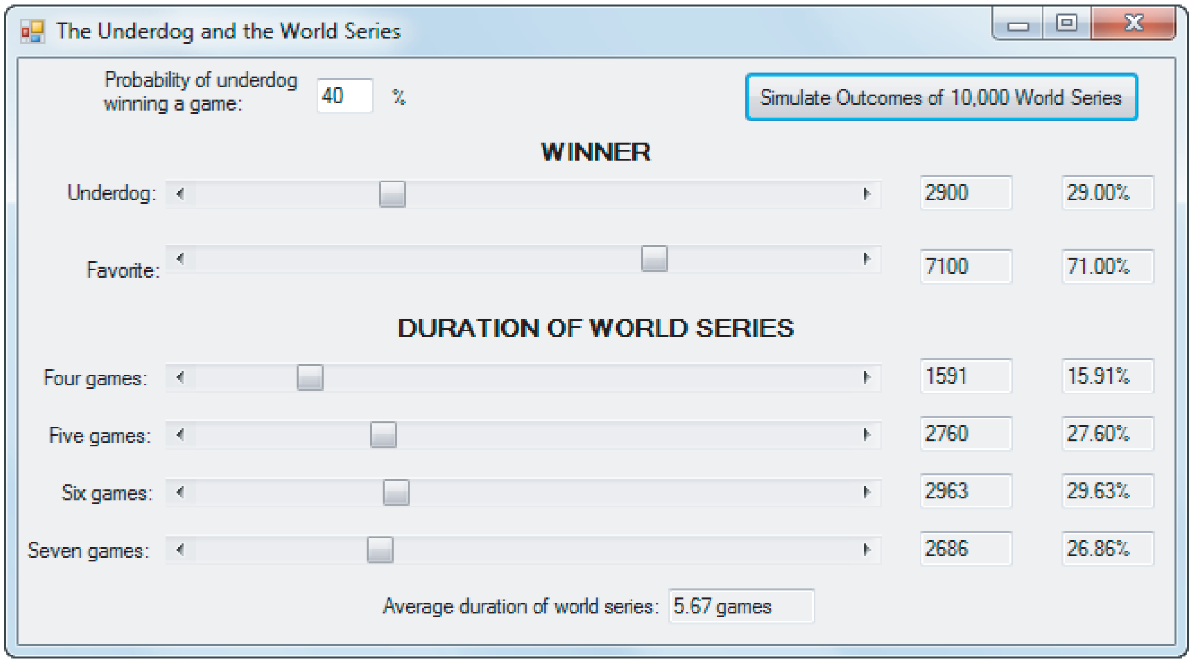
(a) (b)

FIGURE 9.50 **Forms for Programming Project 4.**

5. *The Underdog and the World Series.* What is the probability that the underdog will win the World Series of Baseball? What is the average number of games for a World Series? Write an animated program to answer these questions. For instance, suppose that the underdog is expected to win 40% of the time. We say that the probability of the underdog's winning a game is 40%. (In order for a team to be the underdog, the probability that the team wins a game must be less than 50%.) Figure 9.51 shows that the probability of a 40% underdog's winning the World Series is about 29%, and that such a series would last an average of about 5.67 games. The program should simulate the playing of 10,000 World Series where the underdog has the probability of winning that was entered into the first text box. The values of the horizontal scroll bars should extend from 0 to 10,000 and should be calculated after each series so that the scroll boxes steadily move across the bars. ***Note:*** In order to spare Visual Basic from being overwhelmed with changing the values in the twelve text boxes to the right of the scroll bars too often, just change the values after every ten series. Also, every time the values are changed, execute a Refresh method for each of these text boxes.

6. *Spread of an Epidemic.* A community of 10,000 individuals is exposed to a flu epidemic in which infected individuals are sick for two days and then are immune from the illness. When we first start to observe the epidemic (that is, on day 0), 200 people have had the illness for one day, and 100 people have had the illness for two days. At any time, the rate at which the epidemic is spreading is proportional to the product of the number currently ill and the number susceptible. Specifically, each day

[# of individuals in the first day of the illness] =
    CInt(0.0001735 * [# sick the previous day] * [# susceptible the previous day])


The Underdog and the World Series
Simulate Outcomes of 10,000 World Series
WINNER
DURATION OF WORLD SERIES

Probability of underdog winning a game: 40 %

Underdog: 2900 29.00%

Favorite: 7100 71.00%

Four games: 1591 15.91%

Five games: 2760 27.60%

Six games: 2963 29.63%

Seven games: 2686 26.86%

Average duration of world series: 5.67 games

FIGURE 9.51 A sample run of Programming Project 5.

Write a program that displays successive bar graphs illustrating the progress of the epidemic. When the *Show Day 0* button is clicked on, the bar graph should show the distribution for day 0. See Fig. 9.52. Each time the *Advance One Day* button is clicked on, a bar graph showing the distribution for the next day should appear. Figure 9.53 shows the bar graph after this button has been clicked on three times.

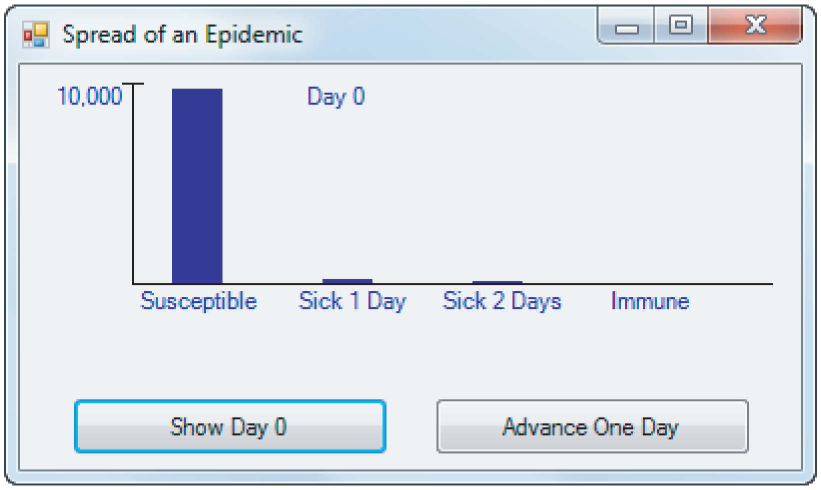


FIGURE 9.52 Initial distribution of epidemic.

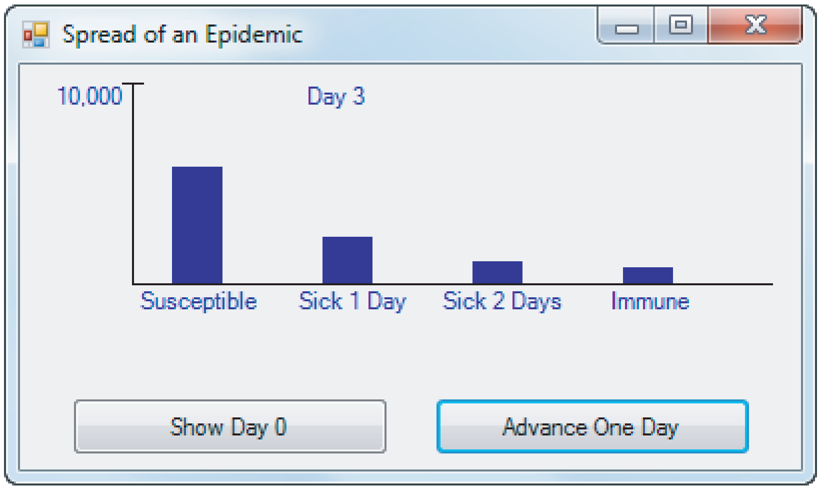


FIGURE 9.53 Distribution on day 3 of epidemic.