43. Rewrite the program in Example 8 without using ElseIf clauses. That is, the task should be carried out with a sequence of simple If blocks.

44. Rewrite the program in Example 8 so that the GPA is validated to be between 2 and 4 before the If block is executed.

Solutions to Practice Problems 4.2

1.
```
If (num < 0) Then
  MessageBox.Show("Number can't be negative.", "Input Error")
  txtNumber.Clear()
  txtNumber.Focus()
Else
  txtSquareRoot.Text = CStr(Math.Sqrt(num))
End If
```

2. The word "hello" will be displayed when (a < b) is true and (c < 5) is also true. That is, it will be displayed when both of these two conditions are true. The clearest way to write the block is

```
If (a < b) And (c < 5) Then
  txtBox.Text = "hello"
End If
```

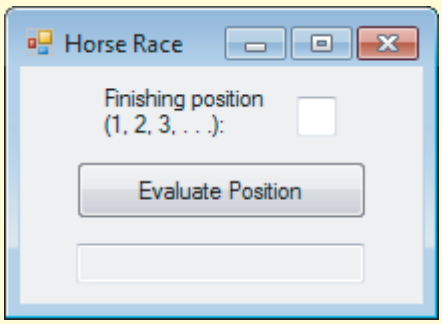
4.3 Select Case Blocks

A Select Case block is an efficient decision-making structure that simplifies choosing among several actions. It avoids complex If constructs. If blocks make decisions based on the truth value of a condition; Select Case choices are determined by the value of an expression called a **selector**. Each possible action is preceded by a clause of the form

```
Case valueList
```

where *valueList* itemizes the values of the selector for which the action should be taken.

Example 1 The following program converts the finishing position in a horse race into a descriptive phrase. After the variable *position* is assigned a value from txtPosition, Visual Basic searches for the first Case clause whose value list contains that value and executes the succeeding statement. If the value of *position* is greater than 5, then the statement following Case Else is executed.

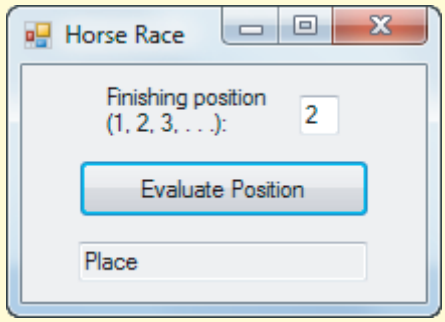


OBJECT	PROPERTY	SETTING
frmRace	Text	Horse Race
lblPosition	AutoSize	False
	Text	Finishing position (1, 2, 3, . . .):
txtPosition		
btnEvaluate	Text	Evaluate Position
txtOutcome	ReadOnly	True

```
Private Sub btnEvaluate_Click(...) Handles btnEvaluate.Click
  Dim position As Integer   'selector
  position = CInt(txtPosition.Text)
```

```
  Select Case position
    Case 1
      txtOutcome.Text = "Win"
    Case 2
      txtOutcome.Text = "Place"
    Case 3
      txtOutcome.Text = "Show"
    Case 4, 5
      txtOutcome.Text = "You almost placed in the money."
    Case Else
      txtOutcome.Text = "Out of the money."
  End Select
End Sub
```

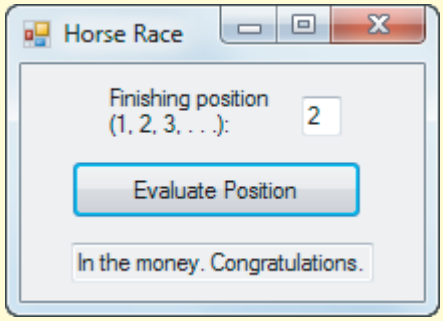
[Run, type 2 into the text box, and click on the button.]



Example 2 In the following variation of Example 1, the value lists specify ranges of values. The first value list provides another way to stipulate the numbers 1, 2, and 3. The second value list covers all numbers from 4 on.

```
Private Sub btnEvaluate_Click(...) Handles btnEvaluate.Click
  'Describe finishing positions in a horse race
  Dim position As Integer
  position = CInt(txtPosition.Text)
  Select Case position
    Case 1 To 3
      txtOutcome.Text = "In the money. Congratulations."
    Case Is >= 4
      txtOutcome.Text = "Not in the money."
  End Select
End Sub
```

[Run, type 2 into the text box, and click on the button.]



General Form of a Select Case Block

A typical form of the Select Case block is

VideoNote
Select Case blocks

```
Select Case selector
  Case valueList 1
    action 1
  Case valueList 2
    action 2
  Case Else
    action of last resort
End Select
```

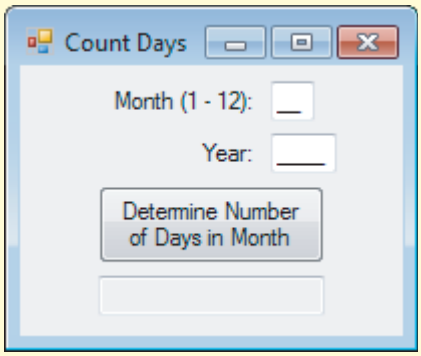
where Case Else (and its action) is optional, and each value list contains one or more of the following types of items:

1. a literal;
2. a variable;
3. an expression;
4. an inequality sign preceded by Is and followed by a literal, variable, or expression;
5. a range expressed in the form `a To b`, where *a* and *b* are literals, variables, or expressions.

Two or more items appearing in the same list must be separated by commas. Each action consists of one or more statements. After the selector is evaluated, Visual Basic looks for the first value-list item including the value of the selector and carries out its associated action. (If the value of the selector appears in two different value lists, only the action associated with the first value list will be carried out.) If the value of the selector does not appear in any of the value lists and there is no Case Else clause, execution of the program will continue with the statement following the Select Case block.

Figure 4.6 contains the flowchart for a Select Case block. The pseudocode for a Select Case block is the same as for the equivalent If block.

Example 3 The following program requests a month and a year, and then displays the number of days in that month. The program uses an If block nested inside a Select Case block. The If block determines whether the year is a leap year. (A year is a leap year if there are 366 days from January 1 of that year to January 1 of the next year.) The value lists come from the rhyme "Thirty days hath September."



OBJECT	PROPERTY	SETTING
frmDays	Text	Count Days
lblMonth:	Text	Month (1–12):
mtbMonth	Mask	00
lblYear	Text	Year:
mtbYear	Mask	0000
btnDetermine	Text	Determine Number of Days in Month
txtOutput	ReadOnly	True

```
Private Sub btnDetermine_Click(...) Handles btnDetermine.Click
  Dim month As Integer = CInt(mtbMonth.Text)
  Dim yr As Integer = CInt(mtbYear.Text)
```

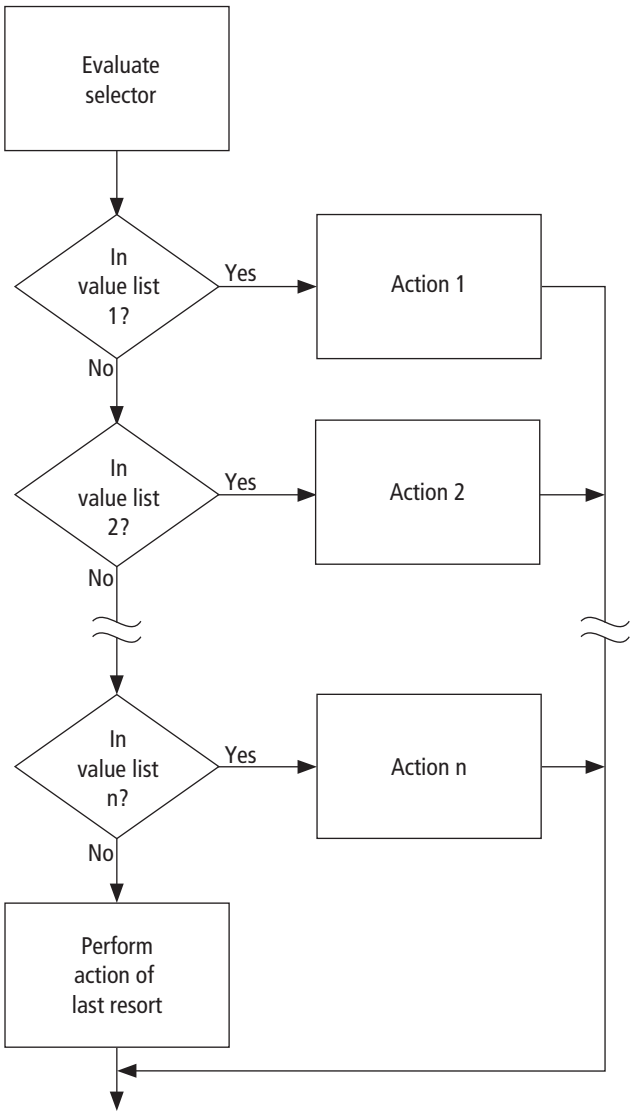
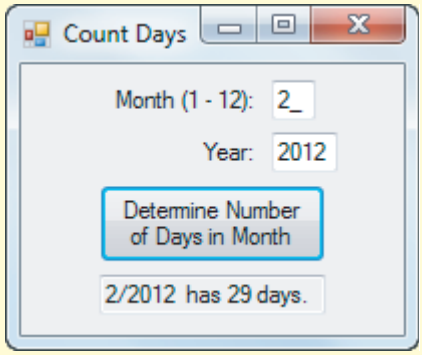


FIGURE 4.6 Flowchart for a select case block.

```
Dim d1, d2 As Date
Dim numberOfDays As Integer
Select Case month
  Case 9, 4, 6, 11   'September, April, June, and November
    numberOfDays = 30
  Case 2    'February
    d1 = CDate("1/1/" & yr)
    d2 = d1.AddYears(1)
    If DateDiff(DateInterval.Day, d1, d2) = 366 Then
      numberOfDays = 29
    Else
      numberOfDays = 28
    End If
  Case Else    'all the rest
    numberOfDays = 31
End Select
```

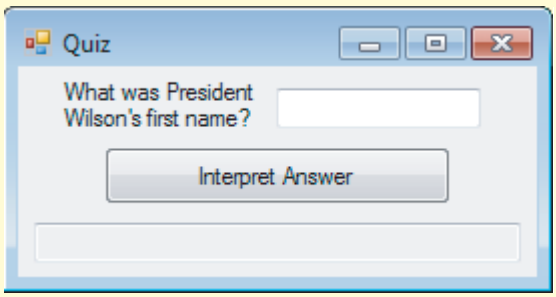
```
  txtOutput.Text = month & "/" & yr & "  has " &
      numberOfDays & " days."
End Sub
```

[Run, enter 2 and 2012 into the text boxes, and click on the button to find the number of days in February, 2012.]



In the three preceding examples, the selector was a numeric variable; however, the selector also can be a string variable, a date variable, or an expression.

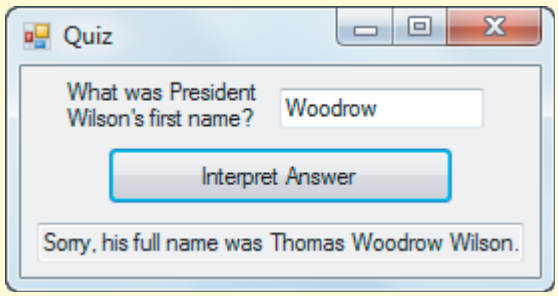
Example 4 The following program has the string variable *firstName* as a selector.



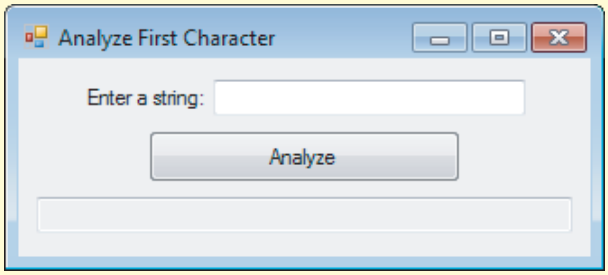
OBJECT	PROPERTY	SETTING
frmQuiz	Text	Quiz
lblQuestion	AutoSize	False
	Text	What was President Wilson's first name?
txtName		
btnInterpret	Text	Interpret Answer
txtReply	ReadOnly	True

```
Private Sub btnInterpret_Click(...) Handles btnInterpret.Click
  'Quiz
  Dim firstName As String
  firstName = txtName.Text.ToUpper
  Select Case firstName
    Case "THOMAS"
      txtReply.Text = "Correct."
    Case "WOODROW"
      txtReply.Text = "Sorry, his full name was " &
                      "Thomas Woodrow Wilson."
    Case "PRESIDENT"
      txtReply.Text = "Are you for real?"
    Case Else
      txtReply.Text = "Nice try, but no cigar."
  End Select
End Sub
```

[Run, type "Woodrow" into the text box, and click on the button.]



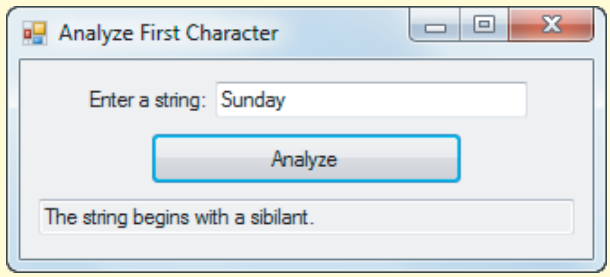
Example 5 The following program has the string selector *anyString*.*Substring* (*0*, *1*). In the sample run, only the first action was carried out, even though the value of the selector was in both of the first two value lists. Visual Basic stops looking as soon as it finds the value of the selector.



OBJECT	PROPERTY	SETTING
frmAnalyze	Text	Analyze First Character
lblEnter	Text	Enter a string:
txtString		
btnAnalyze	Text	Analyze
txtResult	ReadOnly	True

```
Private Sub btnAnalyze_Click(...) Handles btnAnalyze.Click
  'Analyze the first character of a string
  Dim anyString As String
  anyString = txtString.Text.ToUpper
  Select Case anyString.Substring(0, 1)
    Case "S", "Z"
      txtResult.Text = "The string begins with a sibilant."
    Case "A" To "Z"
      txtResult.Text = "The string begins with a nonsibilant."
    Case "0" To "9"
      txtResult.Text = "The string begins with a digit."
    Case Is < "0"
      txtResult.Text = "The string begins with a character of " &
                       "ANSI value less than 48."
    Case Else
      txtResult.Text = "The string begins with :  ;  <  =  > " &
                       " ?  @  [  \  ]  ^  _  or  `  .  "
  End Select
End Sub
```

[Run, type "Sunday" into the text box, and click on the button.]



Comments

1. In a Case clause of the form `Case b To c`, the value of *b* should be less than or equal to the value of *c*. Otherwise, the clause is meaningless.
2. If the word `Is`, which should precede an inequality sign in a value list, is accidentally omitted, the editor will automatically insert it when checking the line.
3. The items in the value list must evaluate to a literal of the same type as the selector. For instance, if the selector evaluated to a string value, as in

```
Dim firstName As String
firstName = txtBox.Text
Select Case firstName
```

then the clause

```
Case firstName.Length
```

would be meaningless.

4. Any variable declared inside an If or Select Case block has **block-level scope**; that is, the variable cannot be referred to by code outside the block.
5. In Appendix D, the section "Stepping through Programs Containing Selection Structures: Chapter 4" uses the Visual Basic debugging tools to trace the flow through a Select Case block.

Practice Problems 4.3

1. Suppose the selector of a Select Case block is the numeric variable *num*. Determine whether each of the following Case clauses is valid.

   (a) `Case 1, 4, Is < 10`
   (b) `Case Is < 5, Is >= 5`
   (c) `Case num = 2` (where the selector *num* is of type Double)

2. Do the following two programs always produce the same output for a whole-number grade from 0 to 100?

```
grade = CInt(txtBox.Text)
Select Case grade
  Case Is >= 90
    txtOutput.Text = "A"
  Case Is >= 60
    txtOutput.Text = "Pass"
  Case Else
    txtOutput.Text = "Fail"
End Select
```

```
grade = CInt(txtBox.Text)
Select Case grade
  Case Is >= 90
    txtOutput.Text = "A"
  Case 60 To 89
    txtOutput.Text = "Pass"
  Case 0 To 59
    txtOutput.Text = "Fail"
End Select
```

EXERCISES 4.3

In Exercises 1 through 6, for each of the responses shown in the parentheses, determine the output displayed in the text box when the button is clicked on.

1.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim age, price As Double
  age = CDbl(InputBox("What is your age?"))
  Select Case age
```

```
    Case Is < 6
      price = 0
    Case 6 To 17
      price = 3.75
    Case Is >= 17
      price = 5
  End Select
  txtOutput.Text = "The price is " & FormatCurrency(price)
End Sub
```

(8.5, 17)

2.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim num As Double
  num = CDbl(InputBox("Enter a number from 5 to 12"))
  Select Case num
    Case 5
      txtOutput.Text = "case 1"
    Case 5 To 7
      txtOutput.Text = "case 2"
    Case 7 To 12
      txtOutput.Text = "case 3"
  End Select
End Sub
```

(7, 5, 11.2)

3.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim age As Integer
  age = CInt(InputBox("Enter age (in millions of years)"))
  Select Case age
    Case Is < 70
      txtOutput.Text = "Cenozoic Era"
    Case Is < 225
      txtOutput.Text = "Mesozoic Era"
    Case Is <= 600
      txtOutput.Text = "Paleozoic Era"
    Case Else
      txtOutput.Text = "?"
  End Select
End Sub
```

(100, 600, 700)

4.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim pres As String
  pres = InputBox("Who was the youngest U.S. president?")
  Select Case pres.ToUpper
    Case "THEODORE ROOSEVELT", "TEDDY ROOSEVELT"
      txtOutput.Text = "Correct. He became president at age 42 " &
                       "when President McKinley was assassinated."
    Case "JFK", "JOHN KENNEDY", "JOHN F. KENNEDY"
      txtOutput.Text = "Incorrect. At age 43, he was the youngest " &
                       "person elected president."
```

```
    Case Else
      txtOutput.Text = "Nope"
  End Select
End Sub
```

(JFK, Teddy Roosevelt)

5.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim message As String, a, b, c As Double
  message = "Analyzing solutions to the quadratic equation " &
            "AX^2 + BX + C = 0.  Enter the value for "
  a = CDbl(InputBox(message & "A"))
  b = CDbl(InputBox(message & "B"))
  c = CDbl(InputBox(message & "C"))
  Select Case (b ^ 2) - (4 * a * c)
    Case Is < 0
      txtOutput.Text = "The equation has no real solutions."
    Case 0
      txtOutput.Text = "The equation has exactly one solution."
    Case Is > 0
      txtOutput.Text = "The equation has two solutions."
  End Select
End Sub
```

(1,2,3; 1,5,1; 1,2,1)

6.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim whatever As Double
  whatever = CDbl(InputBox("Enter a number:"))
  Select Case whatever
    Case Else
      txtOutput.Text = "Hi"
  End Select
End Sub
```

(7, −1)

In Exercises 7 through 12, identify the errors.

7.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim num As Double = 2
  Select Case num
    txtOutput.Text = "Two"
  End Select
End Sub
```

8.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim num1 As Double = 5
  Dim num2 As Double = 2
  Select Case num1
    Case 3 <= num1 <= 10
      txtOutput.Text = "between 3 and 10."
    Case num2 To 5; 4
      txtOutput.Text = "near 5."
  End Select
End Sub
```

9.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim nom As String
  nom = InputBox("What is your name?")
  Select Case nom
    Case nom = "Bob"
      txtOutput.Text = "Hi, Bob."
    Case Else
  End Select
End Sub
```

10.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim word As String = "hello"
  Select Case word.Substring(0,1)
    Case h
      txtOutput.Text = "begins with h."
  End Select
End Sub
```

11.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim fruit As String = "Peach"
  Select Case fruit.ToUpper
    Case Is >= "Peach"
      txtOutput.Text = "Georgia"
    Case "ORANGE To PEACH"
      txtOutput.Text = "Ok"
   End Select
End Sub
```

12.
```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim purchase As Double
  purchase = CDbl(InputBox("Quantity purchased?"))
  Select Case purchase
    Case purchase < 10000
      txtOutput.Text = "Five dollars per item."
    Case Is 10000 To 30000
      txtOutput.Text = "Four dollars per item."
    Case Is > 30000
      txtOutput.Text = "Three dollars per item."
  End Select
End Sub
```

In Exercises 13 through 18, suppose the selector of a Select Case block, *word*, evaluates to a String value. Determine whether the Case clause is valid.

13. `Case "un" & "til"`
14. `Case "hello", Is < "goodbye"`
15. `Case 0 To 9`
16. `Case word <> "No"`
17. `Case "abc".Substring(0, 1)`
18. `Case Is <> "No"`

In Exercises 19 through 22, rewrite the code using a Select Case block.

19.
```
If a = 1 Then
  txtOutput.Text = "one"
Else
  If a > 5 Then
    txtOutput.Text = "two"
  End If
End If
```

20.
```
If a = 1 Then
  lstOutput.Items.Add("lambs")
End If
If ((a <= 3) And (a < 4)) Then
  lstOutput.Items.Add("eat")
End If
If ((a = 5) Or (a > 7)) Then
  lstOutput.Items.Add("ivy")
End If
```

21.
```
If a < 5 Then
  If a = 2 Then
    txtOutput.Text = "yes"
  Else
    txtOutput.Text = "no"
  End If
Else
  If a = 2 Then
   txtOutput.Text = "maybe"
  End If
End If
```

22.
```
If a = 3 Then
  a = 1
End If
If a = 2 Then
  a = 3
End If
If a = 1 Then
  a = 2
End If
```

23. Table 4.5 gives the terms used by the National Weather Service to describe the degree of cloudiness. Write a program that requests the percentage of cloud cover as input and then displays the appropriate descriptor.

TABLE 4.5 Cloudiness descriptors.

Percentage of Cloud Cover	Descriptor
0–30	clear
31–70	partly cloudy
71–99	cloudy
100	overcast

24. Table 4.6 shows the location of books in the library stacks according to their call numbers. Write a program that requests the call number of a book as input and displays the location of the book.

TABLE 4.6 Location of library books.

Call Numbers	Location
100 to 199	basement
200 to 500 and over 900	main floor
501 to 900 except 700 to 750	upper floor
700 to 750	archives

25. Figure 4.7 shows some geometric shapes and formulas for their areas. Write a program that requests the user to select one of the shapes, requests the appropriate lengths, and then gives the area of the figure.

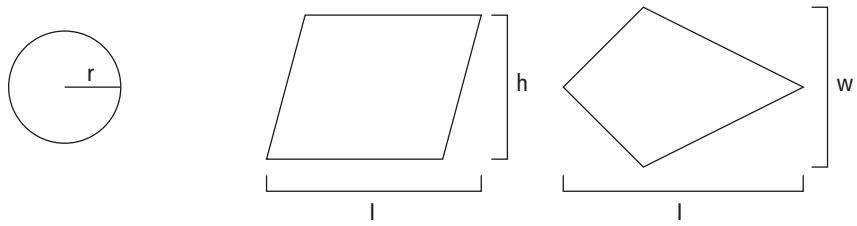


FIGURE 4.7 Areas of geometric shapes.

26. *Break-Even Analysis*. Suppose a certain product sells for a dollars per unit. Then the revenue from selling x units of the product is ax dollars. If the cost of producing each unit of the product is b dollars and the company has overhead costs of c dollars, then the total cost of producing x units of the product is $bx + c$ dollars. (***Note:* Revenue** is the amount of money received from the sale of the product. The values of a, b, and c are called the **marginal revenue**, **marginal cost**, and **fixed cost** respectively. The break-even point is the value of x for which the revenue equals the total cost.) Write a program that requests the marginal revenue, marginal cost, fixed cost, and number of units of the product produced and sold (x) and then displays one of the following three outputs: PROFIT, LOSS, or BREAK EVEN.

27. Write a program that requests an exam score and assigns a letter grade with the scale 90–100 (A), 80–89 (B), 70–79 (C), 60–69 (D), 0–59 (F). (Test the program with the grades 84, 100, and 57.)

VideoNote
Grading system (Homework)

28. Table 4.7 contains information on several states. Write a program that requests a state and category (flower, motto, and nickname) as input and displays the requested information. If the state or category requested is not in the table, the program should so inform the user.

TABLE 4.7 State flowers, nicknames, and mottoes.

State	Flower	Nickname	Motto
California	Golden Poppy	Golden State	Eureka
Indiana	Peony	Hoosier State	Crossroads of America
Mississippi	Magnolia	Magnolia State	By valor and arms
New York	Rose	Empire State	Ever upward

29. IRS informants are paid cash awards based on the value of the money recovered. If the information was specific enough to lead to a recovery, the informant receives 10 percent of the first $75,000, 5 percent of the next $25,000, and 1 percent of the remainder, up to a maximum award of $50,000. Write a program that requests the amount of the recovery as input and displays the award. (Test the program on the amounts $10,000, $125,000, and $10,000,000.) ***Note:*** The source of this formula is *The Book of Inside Information*, Boardroom Books, 1993.

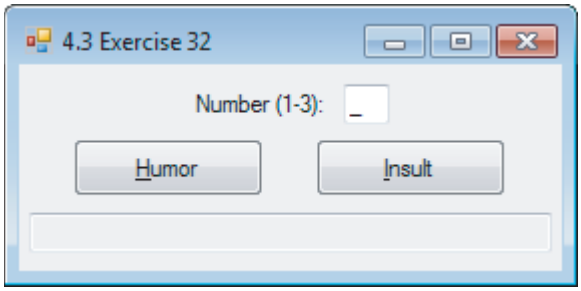
30. Table 4.8 contains the meanings of some abbreviations doctors often use for prescriptions. Write a program that requests an abbreviation and gives its meaning. The user should be informed if the meaning is not in the table.

TABLE 4.8 Physicians' abbreviations.

Abbreviation	Meaning
ac	before meals
ad lib	freely as needed
bid	twice daily
gtt	a drop
hs	at bedtime
qid	four times a day

31. Write a program that, given the last name of one of the six recent presidents beginning with Carter, displays his state and a colorful fact about him. (***Hint:*** The program might need to request further information.) (***Note***: Carter: Georgia; The only soft drink served in the Carter White House was Coca-Cola. Reagan: California; His Secret Service code name was Rawhide. George H. W. Bush: Texas; He celebrated his 85th birthday by parachuting out of an airplane. Clinton: Arkansas; In college he did a good imitation of Elvis Presley. George W. Bush: Texas; He once owned the Texas Rangers baseball team. Obama: Illinois; He was the eighth left-handed president.)

32. Write a programs in which the user enters a number into a masked text box and then clicks on the appropriate button to have either one of three pieces of humor or one of three insults displayed in a text box below the buttons. If the number entered is not between 1 and 3, the masked text box should be cleared. (***Note:*** Some possible bits of humor are "I can resist everything except temptation," "I just heard from Bill Bailey. He's not coming home," and "Adding people to a late software project makes it later." Some possible insults are "How much would you charge to haunt a house?" "I bet you have no more friends than an alarm clock," and "When your IQ rises to 30, sell.")



OBJECT	PROPERTY	SETTING
frmExercise32	Text	4.3 Exercise 32
lblNumber	Text	Number (1–3):
mtbNumber	Mask	0
btnHumor	Text	&Humor
btnInsult	Text	&Insult
txtSentence	ReadOnly	True

Solutions to Practice Problems 4.3

1. (a) Valid. These items are redundant because 1 and 4 are just special cases of `Is < 10`. However, this makes no difference in Visual Basic.

   (b) Valid. These items are contradictory. However, Visual Basic looks at them one at a time until it finds an item containing the value of the selector. The action following this Case clause will always be carried out.

   (c) Not valid. It should be `Case 2`.

2. Yes. However, the program on the right is clearer and therefore preferable.

4.4 Input via User Selections

Programs frequently ask the user to make selections from lists of options. In the questionnaire in Fig. 4.8, students can select their major from a list box, their year from a set of radio buttons, and their computer languages studied from a set of check boxes. After the selections have been made, the user clicks on the *Record Data* button to process the information. The set of radio buttons and set of check boxes are each contained in a group box control. The titles sunk into the tops of the group boxes are *Year* and *Languages Studied*. After selections are made from the three sets of choices, decision structures can be used to process the information. Let's consider the four types of controls in Fig. 4.8 one at a time.

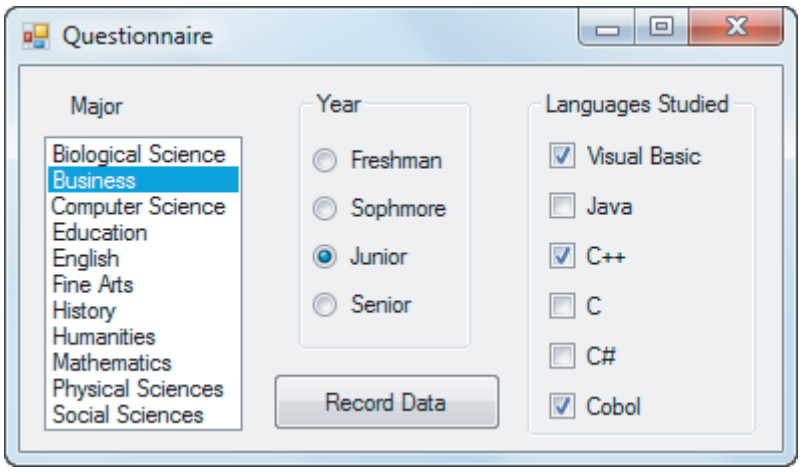


FIGURE 4.8 Selection controls.

Using a List Box for Input

The easiest way to populate a list box with items is to place the items into the list box's String Collection Editor at design time. When you click on the list box's *Tasks* button and then click on *Edit Items* (Fig. 4.9), the String Collection Editor appears. Figure 4.10 shows the String Collection Editor filled with the items from Fig. 4.8. Three ways to fill the String Collection Editor are as follows:

1. Type the items directly into the String Collection Editor.
2. Copy a list of items from any text editor (such as Notepad or Word) with Ctrl + C and paste the list into the String Collection Editor with Ctrl + V.
3. Copy a column of data from a spreadsheet program (such as Excel) and paste it into the String Collection Editor.

When the user clicks on an item at run time, that item is highlighted, and the value of lstBox.Text is that item represented as a string.