blocks of Visual Basic in the early chapters and then puts them together using object-oriented techniques in Chapter 11. Throughout the book, an object-oriented approach is taken whenever feasible.

A Relevant Quote

We end this section with a few paragraphs from *Dirk Gently's Holistic Detective Agency*, by Douglas Adams, Simon & Schuster, 1987:

> "What really is the point of trying to teach anything to anybody?"
>
> This question seemed to provoke a murmur of sympathetic approval from up and down the table.
>
> Richard continued, "What I mean is that if you really want to understand something, the best way is to try and explain it to someone else. That forces you to sort it out in your own mind. And the more slow and dim-witted your pupil, the more you have to break things down into more and more simple ideas. And that's really the essence of programming. By the time you've sorted out a complicated idea into little steps that even a stupid machine can deal with, you've certainly learned something about it yourself. The teacher usually learns more than the pupil. Isn't that true?"

5.5 A Case Study: Weekly Payroll

This case study processes a weekly payroll using the 2009 Employer's Tax Guide. Table 5.3 shows typical data used by a company's payroll office. (***Note:*** A withholding allowance is sometimes referred to as an *exemption*.) These data are processed to produce the information in Table 5.4 that is supplied to each employee along with his or her paycheck. The program should request the data from Table 5.3 for an individual as input and produce output similar to that in Table 5.4.

TABLE 5.3 **Employee data.**

Name	Hourly Wage	Hours Worked	Withholding Allowances	Marital Status	Previous Year-to-Date Earnings
Al Clark	$45.50	38	4	Married	$88,600.00
Ann Miller	$44.00	35	3	Married	$68,200.00
John Smith	$17.95	50	1	Single	$30,604.75
Sue Taylor	$25.50	43	2	Single	$36,295.50

TABLE 5.4 **Payroll information.**

Name	Current Earnings	Yr. to Date Earnings	FICA Tax	Income Tax Wh.	Check Amount
Al Clark	$1,729.00	$90,329.00	$132.27	$163.44	$1,433.29

The items in Table 5.4 should be calculated as follows:

Current Earnings: hourly wage times hours worked (with time-and-a-half after 40 hours)

Year-to-Date Earnings: previous year-to-date earnings plus current earnings

FICA Tax: sum of 6.2% of earnings if part of the first $106,800 of earnings (social security benefits tax) and 1.45% of earnings (Medicare tax)

Federal Income Tax Withheld: subtract $70.19 from the current earnings for each withholding allowance and use Table 5.5 or Table 5.6, depending on marital status

Check Amount: [current earnings] − [FICA taxes] − [income tax withheld]

TABLE 5.5 **2009 Federal income tax withheld for a single person paid weekly.**

Adjusted Weekly Income	Income Tax Withheld
$0 to $138	$0
Over $138 to $200	10% of amount over $138
Over $200 to $696	$6.20 + 15% of amount over $200
Over $696 to $1,279	$80.60 + 25% of amount over $696
Over $1,279 to $3,338	$226.35 + 28% of amount over $1,279
Over $3,338 to $7,212	$802.87 + 33% of amount over $3,338
Over $7,212	$2,081.29 + 35% of amount over $7,212

TABLE 5.6 **2009 Federal income tax withheld for a married person paid weekly.**

Adjusted Weekly Income	Income Tax Withheld
$0 to $303	$0
Over $303 to $470	10% of amount over $303
Over $470 to $1,455	$16.70 + 15% of amount over $470
Over $1,455 to $2,272	$164.45 + 25% of amount over $1,455
Over $2,272 to $4,165	$368.70 + 28% of amount over $2,272
Over $4,165 to $7,321	$898.74 + 33% of amount over $4,165
Over $7,321	$1,940.22 + 35% of amount over $7,321

Designing the Weekly Payroll Program

After the data for an employee from Table 5.3 have been input, the program must compute the five amounts appearing in Table 5.4 and then display the payroll information. These five computations form the basic tasks of the program:

1. Compute current earnings.
2. Compute year-to-date earnings.
3. Compute FICA tax.
4. Compute federal income tax withheld.
5. Compute paycheck amount (that is, take-home pay).

Tasks 1, 2, 3, and 5 are fairly simple. Each involves applying a formula to given data. (For instance, if hours worked are at most 40, then [Current Earnings] = [Hourly Wage] times [Hours Worked].) Thus, we won't break down these tasks any further. Task 4 is more complicated, so we continue to divide it into smaller subtasks.

4. *Compute federal income tax withheld*. First, the employee's pay is adjusted for withholding allowances, and then the amount of income tax to be withheld is computed. The computation of the income tax withheld differs for married and single individuals. Task 4 is, therefore, divided into the following subtasks:

4.1 Compute pay adjusted by withholding allowances.

4.3 Compute income tax withheld for single employee.

4.3 Compute income tax withheld for married employee.

The hierarchy chart in Fig. 5.22 shows the stepwise refinement of the problem.

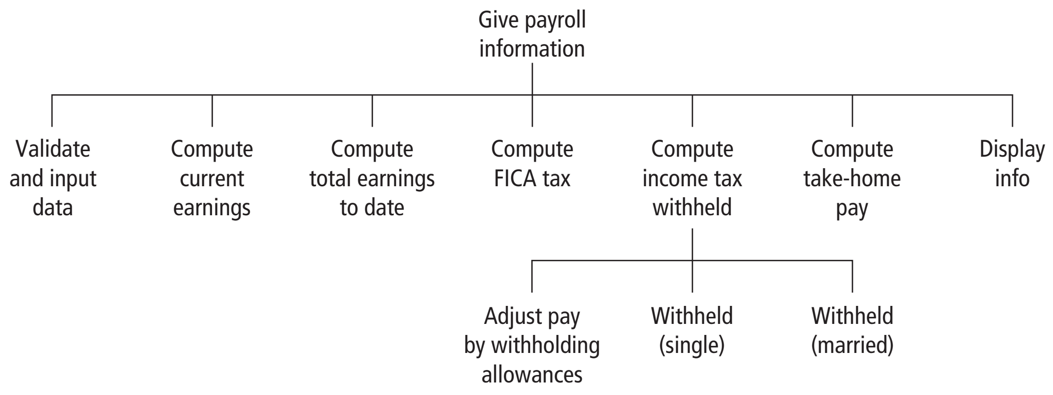


FIGURE 5.22 **Hierarchy chart for the weekly payroll program.**

Pseudocode for the Display Payroll Event Procedure

```
VALIDATE data (Function DataOK)
INPUT employee data (Sub procedure InputData)
COMPUTE CURRENT GROSS PAY (Function Gross_Pay)
COMPUTE TOTAL EARNINGS TO DATE (Function Total_Pay)
COMPUTE FICA TAX (Function FICA_Tax)
COMPUTE INCOME TAX WITHHELD (Function Fed_Tax)
    Adjust pay for withholding allowances
    If employee is single Then
      COMPUTE INCOME TAX WITHHELD (Function TaxSingle)
    Else
      COMPUTE INCOME TAX WITHHELD (Function TaxMarried)
    End If
COMPUTE PAYCHECK AMOUNT (Function Net_Check)
DISPLAY PAYROLL INFORMATION (Sub procedure ShowPayroll)
```

Writing the Weekly Payroll Program

The btnDisplay_Click event procedure calls a sequence of seven procedures. Table 5.7 shows the tasks and the procedures that perform the tasks.

TABLE 5.7 **Tasks and their procedures.**

Task	Procedure
0. Validate and input employee data	DataOK, InputData
1. Compute current earnings.	Gross_Pay
2. Compute year-to-date earnings.	Total_Pay
3. Compute FICA tax.	FICA_Tax
4. Compute federal income tax withheld.	Fed_Tax
4.1 Compute adjusted pay.	Fed_Tax
4.2 Compute amount withheld for single employee.	TaxSingle
4.3 Compute amount withheld for married employee.	TaxMarried
5. Compute paycheck amount.	Net_Check
6. Display payroll information.	ShowPayroll

The Program and the User Interface

Figure 5.23 and Table 5.8 define the user interface for the Weekly Payroll Program. Figure 5.24 shows a sample output.

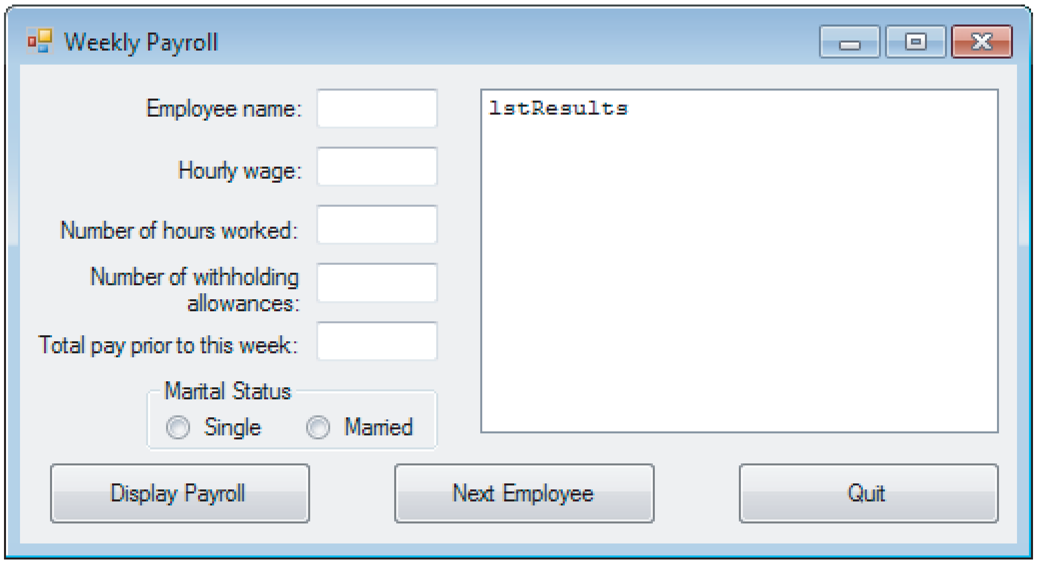
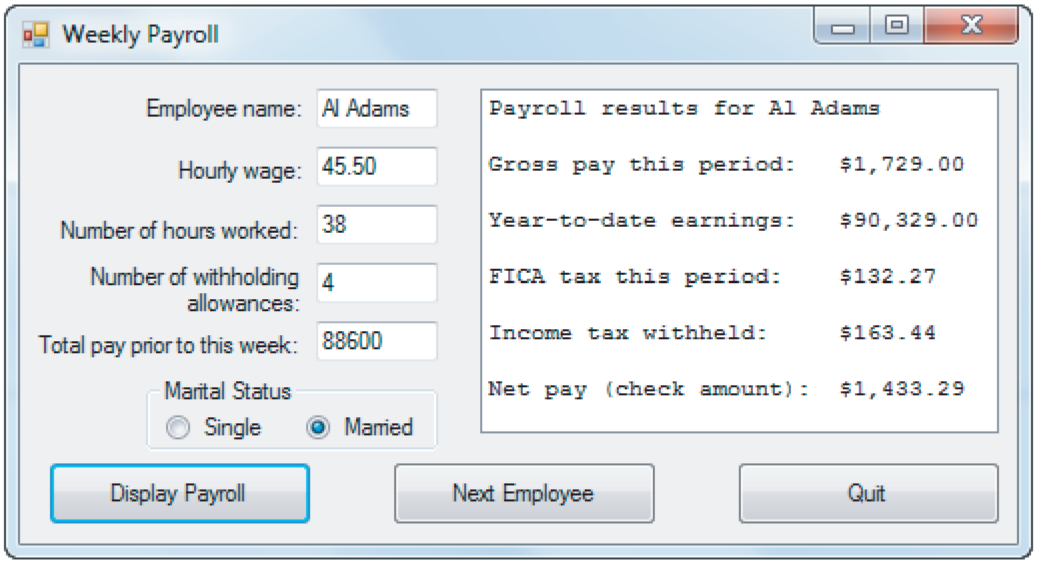


FIGURE 5.23 Form for weekly payroll program.

TABLE 5.8 Objects and initial properties for the weekly payroll program.

Object	Property	Setting
frmPayroll	Text	Weekly Payroll
lblName	Text	Employee name:
txtName		
lblWage	Text	Hourly wage:
txtWage		
lblHours	Text	Number of hours worked:
txtHours		
lblAllowances	AutoSize	False
	Text	Number of withholding allowances:
txtAllowances		
lblPriorPay	Text	Total pay prior to this week:
txtPriorPay		
grpMarital	Text	Marital Status:
radSingle	Text	Single
radMarried	Text	Married
btnDisplay	Text	Display Payroll
btnNext	Text	Next Employee
btnQuit	Text	Quit
lstResults	Font	Courier New



Employee name: Al Adams
Hourly wage: 45.50
Number of hours worked: 38
Number of withholding allowances: 4
Total pay prior to this week: 88600
Marital Status: Single [✓ Married]

```
Payroll results for Al Adams

Gross pay this period:    $1,729.00

Year-to-date earnings:    $90,329.00

FICA tax this period:     $132.27

Income tax withheld:      $163.44

Net pay (check amount):   $1,433.29
```

FIGURE 5.24 Sample output of weekly payroll problem.

```
Private Sub btnDisplay_Click(...) Handles btnDisplay.Click
  Dim empName As String = ""   'Name of employee
  Dim hrWage As Double         'Hourly wage
  Dim hrsWorked As Double      'Hours worked this week
  Dim allowances As Integer    'Number of withholding allowances
  '                             for employee
  Dim prevPay As Double        'Total pay for year excluding this week
  Dim mStatus As String = ""   'Marital status: S for Single; M for Married
  Dim pay As Double            'This week's pay before taxes
  Dim totalPay As Double       'Total pay for year including this week
  Dim ficaTax As Double        'FICA tax for this week
  Dim fedTax As Double         'Federal income tax withheld this week
  Dim check As Double          'Paycheck this week (take-home pay)
  'Verify and obtain data, compute payroll, display results
  If Not DataOK() Then
    Dim msg As String = "At least one piece of requested data is missing" &
                        " or is provided improperly."
    MessageBox.Show(msg)
  Else
    InputData(empName, hrWage, hrsWorked, allowances, prevPay, mStatus) 'Task 0
    pay = Gross_Pay(hrWage, hrsWorked)                                   'Task 1
    totalPay = Total_Pay(prevPay, pay)                                   'Task 2
    ficaTax = FICA_Tax(pay, prevPay, totalPay)                           'Task 3
    fedTax = Fed_Tax(pay, allowances, mStatus)                           'Task 4
    check = Net_Check(pay, ficaTax, fedTax)                              'Task 5
    ShowPayroll(empName, pay, totalPay, ficaTax, fedTax, check)          'Task 6
  End If
End Sub

Private Sub btnNext_Click(...) Handles btnNext.Click
  'Clear all text boxes and radio buttons for next employee's data
  txtName.Clear()
  txtWage.Clear()
  txtHours.Clear()
  txtAllowances.Clear()
  txtPriorPay.Clear()
  radSingle.Checked = False
  radMarried.Checked = False
  lstResults.Items.Clear()
  txtName.Focus()
End Sub

Private Sub btnQuit_Click(...) Handles btnQuit.Click
  Me.Close()
End Sub

Function DataOK() As Boolean
  'Task 0: Validate data
  If (txtName.Text = "") Or (Not IsNumeric(txtWage.Text)) Or
     (Not IsNumeric(txtHours.Text)) Or (Not IsNumeric(txtAllowances.Text)) Or
     (Not IsNumeric(txtPriorPay.Text)) Or
     ((Not radSingle.Checked) And (Not radMarried.Checked)) Then
    Return False
  Else
    Return True
  End If
End Function

Sub InputData(ByRef empName As String, ByRef hrWage As Double,
              ByRef hrsWorked As Double, ByRef allowances As Integer,
              ByRef prevPay As Double, ByRef mStatus As String)
  'Task 0: Input data
  empName = txtName.Text
  hrWage = CDbl(txtWage.Text)
  hrsWorked = CDbl(txtHours.Text)
  allowances = CInt(txtAllowances.Text)
  prevPay = CDbl(txtPriorPay.Text)
  If radMarried.Checked Then
    mStatus = "M"
  Else
    mStatus = "S"
  End If
End Sub

Function Gross_Pay(ByVal hrWage As Double,
                   ByVal hrsWorked As Double) As Double
  'Task 1: Compute weekly pay before taxes
  If hrsWorked <= 40 Then
    Return hrsWorked * hrWage
  Else
    Return 40 * hrWage + (hrsWorked – 40) * 1.5 * hrWage
  End If
End Function

Function Total_Pay(ByVal prevPay As Double, ByVal pay As Double) As Double
  'Task 2: Compute total pay before taxes
  Return prevPay + pay
End Function
```

```
Function FICA_Tax(ByVal pay As Double, ByVal prevPay As Double,
                  ByVal totalPay As Double) As Double
  'Task 3: Compute social security and Medicare tax
  Dim socialSecurity As Double       'Social security tax for this week
  Dim medicare As Double             'Medicare tax for this week
  Dim sum As Double                  'Sum of above two taxes
  Const WAGE_BASE As Double = 106800
  If totalPay <= WAGE_BASE Then
    socialSecurity = 0.062 * pay
  ElseIf prevPay < WAGE_BASE Then
    socialSecurity = 0.062 * (WAGE_BASE - prevPay)
  End If
  medicare = 0.0145 * pay
  sum = socialSecurity + medicare
  Return Math.Round(sum, 2)      'Round to nearest cent
End Function

Function Fed_Tax(ByVal pay As Double, ByVal allowances As Integer,
                 ByVal mStatus As String) As Double
  'Task 4: Compute federal income tax withheld rounded to 2 decimal places.
  Dim adjPay As Double
  Dim tax As Double              'Unrounded federal tax withheld
  adjPay = pay - (70.19 * allowances)     'Task 4.1
  If adjPay < 0 Then
    adjPay = 0
  End If
  If mStatus = "S" Then
    tax = TaxSingle(adjPay)    'Task 4.2
  Else
    tax = TaxMarried(adjPay)   'Task 4.3
  End If
  Return Math.Round(tax, 2)    'Round to nearest cent
End Function

Function TaxSingle(ByVal adjPay As Double) As Double
  'Task 4.2: Compute federal tax withheld for single person.
  Select Case adjPay
    Case 0 To 138
      Return 0
    Case 138 To 200
      Return (0.1 * (adjPay – 138))
    Case 200 To 696
      Return 6.2 + 0.15 * (adjPay – 200)
    Case 696 To 1279
      Return 80.6 + 0.25 * (adjPay – 696)
    Case 1279 To 3338
      Return 226.35 + 0.28 * (adjPay – 1279)
    Case 3338 To 7212
      Return 802.87 + 0.33 * (adjPay – 3338)
    Case Is > 7212
      Return 2081.29 + 0.35 * (adjPay – 7212)
  End Select
End Function
```

```
Function TaxMarried(ByVal adjPay As Double) As Double
  'Task 4.3: Compute federal tax withheld for married person.
  Select Case adjPay
    Case 0 To 303
      Return 0
    Case 303 To 470
      Return 0.1 * (adjPay - 303)
    Case 470 To 1455
      Return 16.7 + 0.15 * (adjPay - 470)
    Case 1455 To 2272
      Return 164.45 + 0.25 * (adjPay - 1455)
    Case 2272 To 4165
      Return 368.7 + 0.28 * (adjPay - 2272)
    Case 4165 To 7321
      Return 898.74 + 0.33 * (adjPay - 4165)
    Case Is > 7321
      Return 1940.22 + 0.35 * (adjPay - 7321)
  End Select
End Function

Function Net_Check(ByVal pay As Double, ByVal ficaTax As Double,
                   ByVal fedTax As Double) As Double
  'Task 5: Compute amount of money paid to employee.
  Dim checkAmount As Double = pay - ficaTax - fedTax
  Return checkAmount
End Function

Sub ShowPayroll(ByVal empName As String, ByVal pay As Double,
                ByVal totalPay As Double, ByVal ficaTax As Double,
                ByVal fedTax As Double, ByVal check As Double)
  'Task 6: Display results of payroll computations
  lstResults.Items.Clear()
  lstResults.Items.Add("Payroll results for " & empName)
  lstResults.Items.Add("")
  lstResults.Items.Add("Gross pay this period:" & "   " &
                       FormatCurrency(pay))
  lstResults.Items.Add("")
  lstResults.Items.Add("Year-to-date earnings:" & "   " &
                       FormatCurrency(totalPay))
  lstResults.Items.Add("")
  lstResults.Items.Add("FICA tax this period:" & "    " &
                       FormatCurrency(ficaTax))
  lstResults.Items.Add("")
  lstResults.Items.Add("Income tax withheld:" & "     " &
                       FormatCurrency(fedTax))
  lstResults.Items.Add("")
  lstResults.Items.Add("Net pay (check amount):" & "  " &
                       FormatCurrency(check))
End Sub
```

Comments

1. In the function FICA_Tax, care has been taken to avoid computing social security benefits tax on income in excess of $106,800 per year. The logic of the program makes sure an employee whose income for the year crosses the $106,800 threshold during a given week is taxed only on the difference between $106,800 and their previous year-to-date earnings.
2. The two functions TaxMarried and TaxSingle use Select Case blocks to incorporate the tax brackets given in Tables 5.5 and 5.6 for the amount of federal income tax withheld. The upper limit of each Case clause is the same as the lower limit of the next Case clause. This ensures that fractional values for *adjPay*, such as 138.50 in the TaxSingle function, will be properly treated as part of the higher salary range.

CHAPTER 5 SUMMARY

1. A *general procedure* is a portion of a program that is accessed by event procedures or other general procedures. The two types of general procedures are *Function procedures* and *Sub procedures*.

2. *Function procedures* are defined in blocks beginning with Function headers and ending with End Function statements. A function is executed by a reference in an expression and returns a value.

3. *Sub procedures* are defined in blocks beginning with Sub headers and ending with End Sub statements. A Sub procedure is accessed (called) by a statement consisting of the name of the procedure.

4. In any procedure, the *arguments* appearing in the calling statement must match the *parameters* of the Sub or Function statement in number, type, and order. They need not have the same names.

5. The *lifetime* of a variable or constant is the period during which it remains in memory. (The value of the variable might change over its lifetime, but it always holds some value.)

6. The *scope* of a variable or constant is the portion of the program that can refer to it. A variable or constant declared inside a Function, Sub, or event procedure has *local* scope and is visible only inside the procedure.

7. *Structured programming* uses modular design to refine large problems into smaller subproblems. Programs are coded using the three logical structures of sequences, decisions, and loops.

CHAPTER 5 PROGRAMMING PROJECTS

1. Write a program to determine a student's GPA. See Fig. 5.25. The user should enter the grade (*A*, *B*, *C*, *D*, or *F*) and the number of credit hours for a course, and then click on the *Record This Course* button. The user should then repeat this process for all his or her courses. After all the courses have been recorded, the user should click on the *Calculate GPA* button. A Function procedure should be used to calculate the quality points for a course. ***Hint:*** This program is similar to Example 5 in Section 5.1.

2. A fast-food vendor sells pizza slices ($1.75), fries ($2.00), and soft drinks ($1.25). Write a program to compute a customer's bill. The program should request the quantity of each item ordered in a Sub procedure, calculate the total cost with a Function procedure, and use a Sub procedure to display an itemized bill. A sample output is shown in Fig. 5.26.