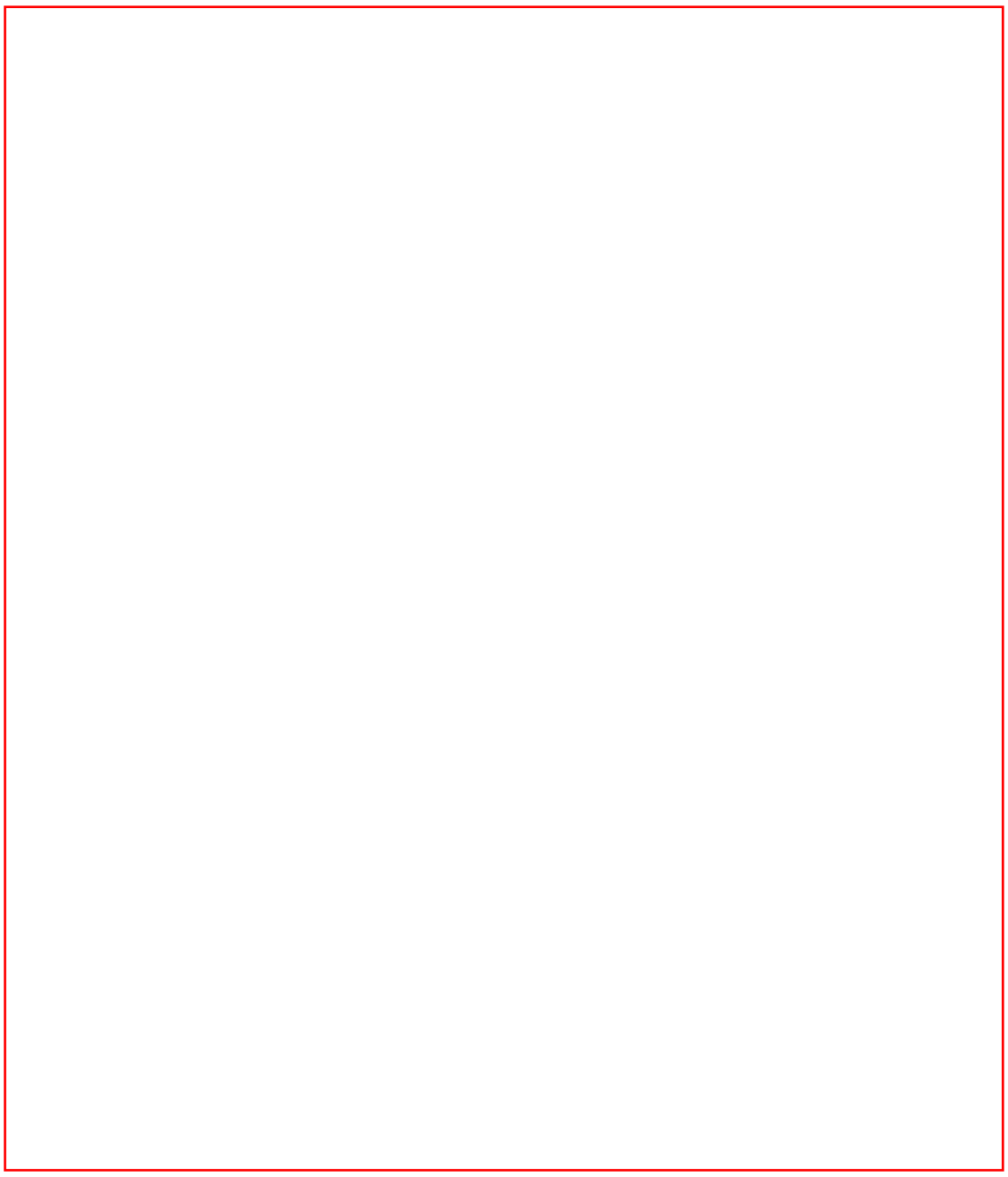

A Visual Basic Program for Gauss-Jordan Elimination

On the next page is Visual Basic code that is designed to run inside Excel and do Gauss-Jordan elimination. Follow these steps:

- Enter the code into Excel by following the instructions on page 32. (the first four bullets)
- Enter an augmented matrix in the upper, left corner of a spreadsheet.
- On the Developer tab in Excel click Macros and run the macro called Gauss_Jordan. A dialog box asks the size of the system. Then the program carries out the steps of the Gauss-Jordan method and replaces the original matrix with the row-reduced matrix.

```
Sub Gauss_Jordan()

   Dim Row As Integer, Col As Integer, K As Integer, N As Integer
   Dim Pivot As Double, Temp As Double, Multiple As Double

   N = InputBox("What is the system's size, N?")          'This problem has N equations in N unknowns.

   For Col = 1 To N                                      'Col is the index of the present (pivot) column.
      MsgBox "Start working on column " & Col            'This loop runs over the columns.
      For Row = Col To N
         If Cells(Row, Col) <> 0 Then GoTo 1             'Find a candidate for the pivot element.
      Next Row                                           'When one is found we can get started.

      MsgBox "ERROR - No unique solution"                'Quick exit:  The present column
      Exit Sub                                           'is full of zeros, so there is no solution, so quit.

1:    If Row <> Col Then                                 'If the candidate pivot is not already on the
         MsgBox "Pivot up row " & Row                    ' diagonal then pivot up its row.
         For K = 1 To N + 1                              'i.e. do ERO # 3.
            Temp = Cells(Row, K)
            Cells(Row, K) = Cells(Col, K)
            Cells(Col, K) = Temp
         Next K
      End If
                                                         'do ERO # 1 to put a 1 in the diagonal (pivot) element.
      Pivot = Cells(Col, Col)

      If Pivot <> 1 Then
         MsgBox "Apply ERO # 1 to row " & Col
         For K = 1 To N + 1                              'Run through all the elements in the row.
            Cells(Col, K) = Cells(Col, K) / Pivot        ' [ ERO#1 ]
         Next K
      Else
         MsgBox "Can skip ERO # 1 on row " & Col
      End If

                                                         'do ERO # 2 to put 0's in each spot above and below the diagonal element.
      For Row = 1 To N
         If Row <> Col Then
            Multiple = Cells(Row, Col)

            If Multiple <> 0 Then
               MsgBox "Apply ERO # 2 to row " & Row
               For K = 1 To N + 1                        'Run through all the elements in the row.
                  Cells(Row, K) = Cells(Row, K) - Multiple * Cells(Col, K)   ' [ ERO#2 ]
               Next K
            Else
               MsgBox "Can skip ERO # 2 on row " & Row
            End If
         End If
      Next Row
   Next Col

   MsgBox "Done!  Answers are in the last column."
End Sub
```

Fig. 2.5 shows an example. (It is in fact Example 2.6.) Note that the code defines subroutine Gauss_Jordan as a Sub rather than as a Function. This means that it has to be run as a macro rather than by typing "=Gauss_Jordan" into a spreadsheet cell. Also note that to stop the display of the message boxes we can just comment them out or delete them.

	A	B	C	D
1	4	8	4	80
2	2	1	-4	7
3	3	-1	2	22

(a)

	A	B	C	D
1	1	0	0	7
2	0	1	0	5
3	0	0	1	3

(b)

Figure 2.5 An example of the use of the Gauss-Jordan program of page 49. (a) The initial augmented matrix. (b) The resulting row-reduced matrix after running the macro with size $N = 3$.