```
Public Sub Gauss(n, ByRef A() As Double, ByRef phi() As Double)
  Dim i, j, k, n_1, p As Integer
  Dim X As Double       'By default, an array begins with zero
  'The extra column is for the Load Vector.
Rem
  'Convert the matrix to a upper triangular matrix
  n_1 = n - 1
  For k = 0 To n_1
    p = k + 1
    For i = p To n
      X = A(i, k) / A(k, k)
      For j = k To n + 1
        A(i, j) = A(i, j) - X * A(k, j)
      Next j
    Next i
  Next k
Rem
  'Back-substitution
  phi(n) = A(n, n + 1) / A(n, n)
  For i = n_1 To 0 Step -1
    Sum = 0#
    For j = i + 1 To n
      Sum = Sum + A(i, j) * phi(j)
    Next j
    phi(i) = (A(i, n + 1) - Sum) / A(i, i)
  Next i
End Sub
'-------------------------------------------------------------------------------
----
Sub checkGauss()
  Dim m, n As Integer
  Dim A() As Double
  Dim phi() As Double
Rem
   ' myArray = Range("A1:F10").Value - Read value from worksheets
Rem  or
   ' Specify matrix value directly
   inMtx = Array(Array(-2.467, 0.5, 1#, 0, -196.7), Array(0.5, -2.267, 0, 1#,
-126.7), _
                 Array(1#, 0#, -4#, 1#, -700#), Array(0#, 1#, 1#, -4#, -600#))
   ReDim A(0 To UBound(inMtx), 0 To UBound(inMtx(1)))
   'UBound(inMtx)    = number of inner arrays
   'UBound(inMtx(1)) = number of elements in inner array
Rem
   n = UBound(A, 1)  'Number of row vectors
   m = UBound(A, 2)  'number of column vectors
   ReDim phi(n)
   ReDim A(n, m)
Rem
   For i = 0 To n
     For j = 0 To m
       A(i, j) = inMtx(i)(j)
     Next j
   Next i
```

```
Rem
   'Call subroutine
   Gauss n, A(), phi()
   For i = 0 To n
     For j = 0 To n
       Worksheets("SOW").Cells(16 + i, j + 3) = A(i, j)
       If Abs(A(i, j)) < 0.001 Then
         Worksheets("SOW").Cells(16 + i, j + 3).NumberFormat = "0"
       Else
         Worksheets("SOW").Cells(16 + i, j + 3).NumberFormat = "0.00"
       End If
     Next j
     Worksheets("SOW").Cells(16 + i, m + 3) = phi(i)
     Worksheets("SOW").Cells(16 + i, m + 3).NumberFormat = "0.00"
   Next i
End Sub
'-------------------------------------------------------------------------------
----
Public Sub triDiag(n, ByRef A() As Double, ByRef phi() As Double)
Rem
  'A(0,i) : AW(i), A(1,i): AP(i), A(2,i): AE(i), A(3,i): d(i)
 Dim i As Integer
 Dim p() As Double
 Dim q() As Double
 ReDim p(n)
 ReDim q(n)
Rem
    p(0) = A(2, 0) / A(1, 0)     ' b(1)/a(1) :: AE(0)/AP(0)
    q(0) = A(3, 0) / A(1, 0)     ' d(1)/a(1)
      For i = 1 To n             ' Forward Elimination
        p(i) = A(0, i) / (A(1, i) - A(2, i - 1) * p(i - 1))
Rem       P(i) :: b(i)/[a(i) - c(i)*P(i-1)]
Rem
        q(i) = (A(3, i) + A(0, i) * q(i - 1)) / (A(1, i) - A(0, i) * p(i - 1))
Rem     Q(i) :: [d(i) + c(i)*Q(i-1)] / [a(i) - c(i) * P(i-1)]
      Next i
Rem
      phi(n) = q(n)        ' Backward Substitution
      For i = n - 1 To 0 Step -1
        phi(i) = p(i) * phi(i + 1) + q(i)
      Next i
End Sub
'-------------------------------------------------------------------------------
----
Sub checkTDMA()
  Dim n, iRow As Integer
  Dim A() As Double
  Dim phi() As Double
   ' myArray = Range("A1:F10").Value - Read value from worksheets
Rem
   ' Specify matrix value directly
   inMtx = Array(Array(0, 1#, 1#, 1#), Array(2.04, 2.04, 2.04, 2.04), _
               Array(1#, 1#, 1#, 0#), Array(40.8, 0.8, 0.8, 200.8))
Rem
```

```
   ReDim A(0 To UBound(inMtx), 0 To UBound(inMtx(1)))
   n = UBound(inMtx(1))
   ReDim phi(n)
   ReDim A(3, n)
Rem
   iRow = 26
   For i = 0 To 3
     For j = 0 To n
       A(i, j) = inMtx(i)(j)
       Worksheets("SOW").Cells(iRow + i, 3 + j) = A(i, j)
     Next j
   Next i
Rem
   'Call subroutine
   triDiag n, A(), phi()
   For i = 0 To n
       Worksheets("SOW").Cells(iRow + i, n + 3) = phi(i)
       If Abs(A(i, 3)) < 0.001 Then
         Worksheets("SOW").Cells(iRow + i, n + 3).NumberFormat = "0"
       Else
         Worksheets("SOW").Cells(iRow + i, n + 3).NumberFormat = "0.00"
       End If
   Next i
End Sub
```