

```
1 'Dan Morgan - VisuMAX Inc.
2 'This program test the speed of two popular forms of sorting algorithms - Insertion
  and Merge sort.
3 'An algorithm analysis shows Insertion sort O(n2) as being rather slow comparied to
  the recursive merge sort O(n log n)
4 'based on asymptotic order as this program shows.
5
6 Public Class frmInsertionSort
7     Inherits System.Windows.Forms.Form
8     Public ICounter As String
9     Public MCounter As String
10    Public History As String
11
12    Windows Form Designer generated code
210
211    Private Sub btnSort_Click(ByVal sender As System.Object, ByVal e As System.
  EventArgs) Handles btnSort.Click
212        Dim DynArray() As Integer
213        Dim i, ArraySize As Integer
214        Dim RandomNum As String
215        Dim rnd As New System.Random
216
217        Call clean()
218
219        RandomNum = "Random Numbers Generated: "
220        Try
221            ArraySize = CInt(txtArraySize.Text) 'Convert our text to an int
222
223            ReDim DynArray(ArraySize - 1) 'Resize the array accordingly
224            For i = 0 To ArraySize 'Loop thru and populate the array.
225                If i = ArraySize Then Exit For
226                DynArray(i) = rnd.Next(1, 1000)
227                RandomNum = RandomNum & vbCrLf & DynArray(i)
228            Next
229
230            RandomNum = RandomNum & vbCrLf & "Random Numbers Sorted:" & vbCr 'Write
  our value generated to our var
231            InsertionSort(DynArray, 0, ArraySize) 'Pass the array to our insertion
  sort algorithm routine
232
233            For i = 0 To ArraySize 'Loop thru and print out each index in the array
234                If i = ArraySize Then Exit For
235                RandomNum = RandomNum & vbCrLf & DynArray(i)
236            Next
237            txtArrayList.Text = RandomNum 'Fill our text box w/ the results
238            txtHistory.Text = History
239        Catch ex As Exception
240            MsgBox("You must enter a number.", MsgBoxStyle.Critical, "Oops!")
241            txtArraySize.Text = ""
242        End Try
243
244    End Sub
245
246    'Insertion Sort algorithm which will move thru the array from the bottom (0) and
  move each indexes
247    'value down and insert it into the correct position.
248    Public Sub InsertionSort(ByVal Arr() As Integer, ByVal LoBound As Integer, ByVal
  HiBound As Integer)
249        Dim i, j, k, Key As Integer
250        Dim T1, T2 As Date
251        Dim TS As TimeSpan
252
253        T1 = Now 'Init the timer
254
255        'Set up the outer for loop
256        For i = LoBound To HiBound - 1 'All arrays in .Net are Zero based
257            Key = Arr(i) 'Grab the value to be inserted
```

```
258
259     For j = i - 1 To LoBound Step -1      'Loop Until we find its spot
260         If Key >= Arr(j) Then             'We found it. Exit the loop
261             ICounter += 1                 'Count the iteration
262             Exit For
263         End If
264
265     Arr(j + 1) = Arr(j)                   'Step over 1 position in the array
266
267     If chkHistory.Checked = True Then     'Print out a history of the Array
268         For k = 0 To HiBound
269             If k = HiBound Then Exit For
270             History = History & Arr(k) & ", "
271         Next
272         History = History & vbCrLf
273     End If
274
275     Next j
276
277     Arr(j + 1) = Key                       'We found a home.
278
279     If chkHistory.Checked = True Then     'Print out a history of the Array
280         For k = 0 To HiBound
281             If k = HiBound Then Exit For
282             History = History & Arr(k) & ", "
283         Next
284         History = History & vbCrLf
285     End If
286     lblICounter.Text = CStr(ICounter)
287
288     Next i
289
290     T2 = Now                               'Array is sorted. Stop the timer.
291
292     TS = T2.Subtract(T1)                   'Get the timer result
293     lblTimer.Text = TS.Milliseconds.ToString 'Write it to our label
294
295
296     MsgBox("Number stored in Key " & Key & " >= Arr(j)" & Arr(j), MsgBoxStyle.  ✎
297     OKOnly, "Array")
298
299     Private Sub btnMergeSort_Click(ByVal sender As System.Object, ByVal e As System.  ✎
300     EventArgs) Handles btnMergeSort.Click
301         Dim DynArray() As Integer
302         Dim i, ArraySize As Integer
303         Dim RandomNum As String
304         Dim rnd As New System.Random
305
306         Call clean()
307
308         RandomNum = "Random Numbers Generated: "
309         Try
310             ArraySize = CInt(txtArraySize.Text) 'Convert our text to an int
311
312             ReDim DynArray(ArraySize - 1) 'Resize the array accordingly
313             For i = 0 To ArraySize 'Loop thru and populate the array with random  ✎
314                 values.
315                 If i = ArraySize Then Exit For
316                 DynArray(i) = rnd.Next(1, 1000)
317                 RandomNum = RandomNum & vbCrLf & DynArray(i)
318             Next
319
320             RandomNum = RandomNum & vbCrLf & "Random Numbers Sorted:" & vbCrLf 'Write  ✎
321             our value generated to our var
322             MergeSort(DynArray) 'Pass the array to our Merge sort algorithm routine
323
324         End Try
325     End Sub
```

```
321     For i = 0 To ArraySize 'Loop thru and print out each index in the array
322         If i = ArraySize Then Exit For
323         RandomNum = RandomNum & vbCrLf & DynArray(i)
324     Next
325     txtArrayList.Text = RandomNum 'Fill our text box w/ the results
326     txtHistory.Text = History
327     Catch ex As Exception
328         MsgBox("You must enter a number.", MsgBoxStyle.Critical, "Oops!")
329     txtArraySize.Text = ""
330
331 End Try
332
333 End Sub
334
335 Public Sub MergeSort(ByRef lngArray() As Integer)
336     Dim arrTemp() As Integer
337     Dim iSegSize As Integer
338     Dim iLBound As Integer
339     Dim iUBound As Integer
340     Dim NewBound As Integer
341     Dim T1, T2 As Date
342     Dim TS As TimeSpan
343
344     T1 = Now 'Init the timer
345
346     iLBound = LBound(lngArray)
347     iUBound = UBound(lngArray)
348
349     NewBound = iUBound - iLBound
350
351     ReDim arrTemp(NewBound)
352
353     iSegSize = 1
354     Do While iSegSize < iUBound - iLBound
355
356         MCounter += 1
357
358         'Merge from A to B
359         InnerMergePass(lngArray, arrTemp, iLBound, iUBound, iSegSize)
360         iSegSize = iSegSize + iSegSize
361
362         'Merge from B to A
363         InnerMergePass(arrTemp, lngArray, iLBound, iUBound, iSegSize)
364         iSegSize = iSegSize + iSegSize
365
366     Loop
367
368     T2 = Now 'Array is sorted. Stop the timer.
369
370     TS = T2.Subtract(T1) 'Get the timer result
371     lblTimer.Text = TS.Milliseconds.ToString 'Write it to our label
372 End Sub
373
374 Private Sub InnerMergePass(ByRef lngSrc() As Integer, ByRef lngDest() As Integer,
375     ByVal iLBound As Integer, ByVal iUBound As Integer, ByVal iSegSize As Integer)
376     Dim iSegNext As Integer
377     iSegNext = iLBound
378
379     Do While iSegNext <= iUBound - (2 * iSegSize)
380         'Merge 2 segments from src to dest
381         InnerMerge(lngSrc, lngDest, iSegNext, iSegNext + iSegSize - 1, iSegNext +
382             iSegSize + iSegSize - 1)
383         iSegNext = iSegNext + iSegSize + iSegSize
384     Loop
385     'Fewer than 2 full segments remain
```

```
386     If iSegNext + iSegSize < iUBound Then
387         '2 segs remain
388         InnerMerge(lngSrc, lngDest, iSegNext, iSegNext + iSegSize - 1, iUBound)
389     Else
390         '1 seg remains, just copy it
391         For iSegNext = iSegNext To iUBound
392             lngDest(iSegNext) = lngSrc(iSegNext)
393         Next iSegNext
394     End If
395
396 End Sub
397
398 Private Sub InnerMerge(ByRef lngSrc() As Integer, ByRef lngDest() As Integer,
399     ByVal iStartFirst As Integer, ByVal iEndFirst As Integer, ByVal iEndSecond As
400     Integer)
401     Dim iFirst As Integer
402     Dim iSecond As Integer
403     Dim iResult As Integer
404     Dim iOuter As Integer
405
406     iFirst = iStartFirst
407     iSecond = iEndFirst + 1
408     iResult = iStartFirst
409
410     Do While (iFirst <= iEndFirst) And (iSecond <= iEndSecond)
411
412         'Select the smaller value and place in the output
413         'Since the subarrays are already sorted, only one comparison is needed
414         If lngSrc(iFirst) <= lngSrc(iSecond) Then
415             lngDest(iResult) = lngSrc(iFirst)
416             History = History & "," & lngDest(iResult) 'Keep History
417             iFirst = iFirst + 1
418         Else
419             lngDest(iResult) = lngSrc(iSecond)
420             History = History & "," & lngDest(iResult) 'Keep History
421             iSecond = iSecond + 1
422         End If
423
424         iResult = iResult + 1
425     Loop
426
427     'Take care of any leftover values
428     If iFirst > iEndFirst Then
429         'Got some leftover seconds
430         For iOuter = iSecond To iEndSecond
431             lngDest(iResult) = lngSrc(iOuter)
432             History = History & "," & lngDest(iResult) 'Keep History
433             iResult = iResult + 1
434         Next iOuter
435     Else
436         'Got some leftover firsts
437         For iOuter = iFirst To iEndFirst
438             lngDest(iResult) = lngSrc(iOuter)
439             History = History & "," & lngDest(iResult) 'Keep History
440             iResult = iResult + 1
441         Next iOuter
442     End If
443 End Sub
444
445 Private Sub btnTest_Click(ByVal sender As System.Object, ByVal e As System.
446     EventArgs) Handles btnTest.Click
447     Dim DynArray() As Integer = {40, 70, 60, 80, 50, 30}
448     Dim i, ArraySize As Integer
449     Dim RandomNum As String
450     Dim rnd As New System.Random
```

```
450
451     RandomNum = "Random Numbers Generated: "
452
453     ArraySize = 6
454
455     For i = 0 To ArraySize 'Loop thru and populate the array.
456         If i = 5 Then Exit For
457         RandomNum = RandomNum & vbCrLf & DynArray(i)
458     Next
459
460     RandomNum = RandomNum & vbCrLf & "Random Numbers Sorted:" & vbCrLf 'Write our
value generated to our var
461     InsertionSort(DynArray, 0, ArraySize) 'Pass the array to our insertion sort
algorithm routine
462
463     RandomNum = RandomNum & vbCrLf & "Sorted Numbers"
464     For i = 0 To ArraySize 'Loop thru and print out each index in the array
465         If i = ArraySize Then Exit For
466         RandomNum = RandomNum & vbCrLf & DynArray(i)
467     Next
468     txtArrayList.Text = RandomNum & vbCrLf 'Fill our text box w/ the results
469     txtHistory.Text = History
470 End Sub
471
472 Public Sub clean()
473     txtArrayList.Text = ""
474     txtHistory.Text = ""
475     lblICounter.Text = ""
476     lblTimer.Text = ""
477     History = ""
478
479 End Sub
480 End Class
481
```