Sortieralgorithmen: QuickSort

Quellcode

```
Module Module1
'liste mit 15 Elementen, welches später sortiert werden soll deklarieren
Dim list_to_be_sorted(15) As Integer
'Hilfsroutine um ein Integer-Array in der Konsole auszugeben
Sub print_list(ByRef liste() As Integer)
For Each element In liste
Console.Write(element.ToString & " ")
Next
Console.WriteLine("-")
End Sub
'Hilfsroutine um eine Liste von Integern in der Konsole auszugeben
Sub print_list(ByRef liste As List(Of Integer))
For Each element In liste
Console.Write(element.ToString & " ")
Next
Console.WriteLine("-")
End Sub
'Methode um ein liste mit Zufallswerten zu initialisieren
Sub initialize_list(ByRef liste() As Integer)
'Zufallsgenerator initialisieren
Dim r As New Random(System.DateTime.Now.Millisecond)
'Jedem Element der Schleife einen zufälligen Wert zuweisen
For i As Integer = 0 To liste.Length - 1
'Zufallszahl zwischen 0 und 10.000 erzeugen und zuweisen
liste(i) = r.Next(0, 10000)
Next
End Sub
'Hilfsmethode um zwei stellen im liste zu vertauschen
Sub swap(ByRef x As Integer, ByRef y As Integer)
Dim tmp As Integer
tmp = x
x = y
y = tmp
End Sub
'VB Implementierung des BubbleSort-Algorithmus wie er auf Wikipedia beschrieben ist:
http://de.wikipedia.org/wiki/Bubblesort
Sub bubblesort(ByRef liste() As Integer)
Dim n As Integer = liste.Length
Dim swapped As Boolean
Do
swapped = False
For i As Integer = 0 To n - 2
If (liste(i) > liste(i + 1)) Then
swap(liste(i), liste(i + 1))
swapped = True
End If
Next
n = n - 1
Loop While n > 1
End Sub
'VB Implementierung des Insertionsort-Algorithmus wie er auf Wikipedia beschrieben ist:
http://de.wikipedia.org/wiki/Insertionsort
Sub insertionsort(ByRef liste() As Integer) '
```

```
Dim val, j As Integer
For i As Integer = 1 To liste.Length - 1
val = liste(i)
j = i
While (j > 0 AndAlso liste(j - 1) > val)
liste(j) = liste(j - 1)
j = j - 1
liste(j) = val
End While
Next
End Sub
' MergeSort implementation
Sub MergeSort(ByRef szArray As List(Of Integer), ByVal nLower As Integer, ByVal nUpper As Integer)
' Check for array size
Dim szSwap As String
If (nUpper - nLower) = 1 Then
' Swap strings if necessary
If szArray(nLower).CompareTo(szArray(nUpper)) > 0 Then
szSwap = szArray(nLower)
szArray(nLower) = szArray(nUpper)
szArray(nUpper) = szSwap
End If
ElseIf (nUpper - nLower) > 1 Then
' Sort each half and merge
MergeSort(szArray, nLower, (nLower + nUpper) / 2)
MergeSort(szArray, (nLower + nUpper) / 2 + 1, nUpper)
Merge(szArray, nLower, (nLower + nUpper) / 2 + 1, nUpper)
End If
End Sub
Sub Merge(ByRef szArray As List(Of Integer), ByVal nLower As Integer, ByVal nMiddle As Integer, ByVal nUpper
As Integer)
Dim nIndex As Integer
Dim szBuffer As New List(Of Integer)()
Dim i, j As Integer
i = nLower
j = nMiddle
While i < nMiddle And j <= nUpper
If (szArray(i).CompareTo(szArray(j)) < 0) Then
szBuffer.Add(szArray(i))
i = i + 1
Else
szBuffer.Add(szArray(j))
j = j + 1
End If
End While
While i < nMiddle
szBuffer.Add(szArray(i))
i = i + 1
End While
While j <= nUpper
szBuffer.Add(szArray(j))
j = j + 1
End While
nIndex = 0
For i = nLower To nUpper
szArray(i) = szBuffer(nIndex)
nIndex = nIndex + 1
Next i
```

```
130.  End Sub
131.  Public Function QuickSort(ByVal list As List(Of Integer)) As List(Of Integer)
132.  If list.Count < 1 Then Return list
133.  Dim Smaller As New List(Of Integer)
134.  Dim Larger As New List(Of Integer)
135.  Dim pt As Integer = 0
136.  QuickSort = New List(Of Integer)
137.  Dim pv As Integer = list(Int(Rnd() * list.Count))
138.  For i As Integer = 0 To list.Count - 1
139.  Select Case list(i)
140.  Case Is = pv
141.  pt += 1
142.  Case Is < pv
143.  Smaller.Add(list(i))
144.  Case Is > pv
145.  Larger.Add(list(i))
146.  End Select
147.  Next
148.  QuickSort.AddRange(QuickSort(Smaller))
149.  While pt > 0
150.  QuickSort.Add(pv)
151.  pt -= 1
152.  End While
153.  QuickSort.AddRange(QuickSort(Larger))
154.  End Function
155.  Sub Main()
156.  Dim al As List(Of Integer) = New List(Of Integer)
157.  initialize_list(list_to_be_sorted)
158.  al.AddRange(list_to_be_sorted)
159.  print_list(list_to_be_sorted)
160.  'bubblesort(liste_to_be_sorted)
161.  'insertionsort(liste_to_be_sorted)
162.  'MergeSort(al, 0, list_to_be_sorted.Length - 1)
163.  al = QuickSort(al)
164.  print_list(al)
165.  'print_list(list_to_be_sorted)
166.  Console.ReadKey()
167.  End Sub
168.  End Module
```

Alles anzeigen

Ich werde hier nicht auf die Spezifika des Algorithmus eingehen, da das sprachenunabhängig wäre und eher in die Rubrik Allgemein gehört. Es sei an dieser Stelle auf den Wikipedia-Artikel verwiesen, der eigentlich alles Nötige erklärt.