

Sortieralgorithmen: QuickSort

Quellcode

```
1. Module Module1
2. 'liste mit 15 Elementen, welches später sortiert werden soll deklarieren
4. Dim list_to_be_sorted(15) As Integer
6. 'Hilfsroutine um ein Integer-Array in der Konsole auszugeben
7. Sub print_list(ByRef liste() As Integer)
8. For Each element In liste
9. Console.Write(element.ToString & " ")
10. Next
11. Console.WriteLine("-")
12. End Sub
13. 'Hilfsroutine um eine Liste von Integern in der Konsole auszugeben
15. Sub print_list(ByRef liste As List(Of Integer))
16. For Each element In liste
17. Console.Write(element.ToString & " ")
18. Next
19. Console.WriteLine("-")
20. End Sub
22. 'Methode um ein liste mit Zufallswerten zu initialisieren
23. Sub initialize_list(ByRef liste() As Integer)
24. 'Zufallsgenerator initialisieren
25. Dim r As New Random(System.DateTime.Now.Millisecond)
26. 'Jedem Element der Schleife einen zufälligen Wert zuweisen
27. For i As Integer = 0 To liste.Length - 1
28. 'Zufallszahl zwischen 0 und 10.000 erzeugen und zuweisen
29. liste(i) = r.Next(0, 10000)
30. Next
31. End Sub
32. 'Hilfsmethode um zwei stellen im liste zu vertauschen
34. Sub swap(ByRef x As Integer, ByRef y As Integer)
35. Dim tmp As Integer
36. tmp = x
37. x = y
38. y = tmp
39. End Sub
40. 'VB Implementierung des BubbleSort-Algorithmus wie er auf Wikipedia beschrieben ist:
    http://de.wikipedia.org/wiki/Bubblesort
42. Sub bubblesort(ByRef liste() As Integer)
43. Dim n As Integer = liste.Length
44. Dim swapped As Boolean
45. Do
46. swapped = False
47. For i As Integer = 0 To n - 2
48. If (liste(i) > liste(i + 1)) Then
49. swap(liste(i), liste(i + 1))
50. swapped = True
51. End If
52. Next
53. n = n - 1
54. Loop While n > 1
55. End Sub
56. 'VB Implementierung des Insertionsort-Algorithmus wie er auf Wikipedia beschrieben ist:
    http://de.wikipedia.org/wiki/Insertionsort
58. Sub insertionsort(ByRef liste() As Integer) '
```

```

59. Dim val, j As Integer
60. For i As Integer = 1 To liste.Length - 1
61. val = liste(i)
62. j = i
63. While (j > 0 AndAlso liste(j - 1) > val)
64. liste(j) = liste(j - 1)
65. j = j - 1
66. liste(j) = val
67. End While
68. Next
69. End Sub
70. ' MergeSort implementation
73. Sub MergeSort(ByRef szArray As List(Of Integer), ByVal nLower As Integer, ByVal nUpper As Integer)
74. ' Check for array size
76. Dim szSwap As String
77. If (nUpper - nLower) = 1 Then
78. ' Swap strings if necessary
80. If szArray(nLower).CompareTo(szArray(nUpper)) > 0 Then
81. szSwap = szArray(nLower)
82. szArray(nLower) = szArray(nUpper)
83. szArray(nUpper) = szSwap
84. End If
86. ElseIf (nUpper - nLower) > 1 Then
87. ' Sort each half and merge
89. MergeSort(szArray, nLower, (nLower + nUpper) / 2)
90. MergeSort(szArray, (nLower + nUpper) / 2 + 1, nUpper)
91. Merge(szArray, nLower, (nLower + nUpper) / 2 + 1, nUpper)
92. End If
93. End Sub
94. Sub Merge(ByRef szArray As List(Of Integer), ByVal nLower As Integer, ByVal nMiddle As Integer, ByVal nUpper
    As Integer)
95. Dim nIndex As Integer
100. Dim szBuffer As New List(Of Integer)()
101. Dim i, j As Integer
103. i = nLower
104. j = nMiddle
105. While i < nMiddle And j <= nUpper
106. If (szArray(i).CompareTo(szArray(j)) < 0) Then
107. szBuffer.Add(szArray(i))
108. i = i + 1
109. Else
110. szBuffer.Add(szArray(j))
111. j = j + 1
112. End If
113. End While
116. While i < nMiddle
117. szBuffer.Add(szArray(i))
118. i = i + 1
119. End While
120. While j <= nUpper
121. szBuffer.Add(szArray(j))
122. j = j + 1
123. End While
124. nIndex = 0
126. For i = nLower To nUpper
127. szArray(i) = szBuffer(nIndex)
128. nIndex = nIndex + 1
129. Next i

```

```

130. End Sub
132. Public Function QuickSort(ByVal list As List(Of Integer)) As List(Of Integer)
135. If list.Count < 1 Then Return list
136. Dim Smaller As New List(Of Integer)
137. Dim Larger As New List(Of Integer)
138. Dim pt As Integer = 0
139. QuickSort = New List(Of Integer)
140. Dim pv As Integer = list(Int(Rnd() * list.Count))
141. For i As Integer = 0 To list.Count - 1
142. Select Case list(i)
143. Case Is = pv
144. pt += 1
145. Case Is < pv
146. Smaller.Add(list(i))
147. Case Is > pv
148. Larger.Add(list(i))
149. End Select
150. Next
151. QuickSort.AddRange(QuickSort(Smaller))
152. While pt > 0
153. QuickSort.Add(pv)
154. pt -= 1
155. End While
156. QuickSort.AddRange(QuickSort(Larger))
157. End Function
159. Sub Main()
160. Dim al As List(Of Integer) = New List(Of Integer)
161. initialize_list(list_to_be_sorted)
162. al.AddRange(list_to_be_sorted)
163. print_list(list_to_be_sorted)
166. 'bubblesort(liste_to_be_sorted)
167. 'insertionsort(liste_to_be_sorted)
168. 'MergeSort(al, 0, list_to_be_sorted.Length - 1)
169. al = QuickSort(al)
170. print_list(al)
172. 'print_list(list_to_be_sorted)
173. Console.ReadKey()
175. End Sub
176. 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.