Datenstrukturen und Algorithmen

Übungsblatt 1

Robert Gall, 3408913, robert@gall.cc
Chong Shen, 3111514, shenchong123@yahoo.com
David Lieb, 3408382, st161483@stud.uni-stuttgart.de

Aufgabe 1

(a) sequenzielle Suche

Schritt 1: [11] 22 45 47 66 51 67 72 79 80 81 86 87 88 97
Schritt 2: 11 [22] 45 47 66 51 67 72 79 80 81 86 87 88 97
Schritt 3: 11 22 [45] 47 66 51 67 72 79 80 81 86 87 88 97
Schritt 4: 11 22 45 [47] 66 51 67 72 79 80 81 86 87 88 97
Schritt 5: 11 22 45 47 [66] 51 67 72 79 80 81 86 87 88 97
Schritt 6: 11 22 45 47 66 [51] 67 72 79 80 81 86 87 88 97

binäre Suche

Schritt 1: [11 22 45 47 66 51 67 [72] 79 80 81 86 87 88 97]
Schritt 2: [11 22 45 [47] 66 51 67] 72 [79 80 81 86 87 88 97]
Schritt 3: [11 22 45] 47 [66 [51] 67] 72 79 80 81 86 87 88 97

(b) sequenzielle Suche

Schritt 1: [12] 15 21 22 24 31 40 48 55 59 71 88 91 96 97
Schritt 2: 12 [15] 21 22 24 31 40 48 55 59 71 88 91 96 97
Schritt 3: 12 15 [21] 22 24 31 40 48 55 59 71 88 91 96 97
Schritt 4: 12 15 21 [22] 24 31 40 48 55 59 71 88 91 96 97
Schritt 5: 12 15 21 22 [24] 31 40 48 55 59 71 88 91 96 97
Schritt 6: 12 15 21 22 24 [31] 40 48 55 59 71 88 91 96 97

binäre Suche

Schritt 1: [12 15 21 22 24 31 40 [48] 55 59 71 88 91 96 97]
Schritt 2: [12 15 21 [22] 24 31 40] 48 [55 59 71 88 91 96 97]
Schritt 3: [12 15 21] [22] [24 31 40] 48 55 59 71 88 91 96 97
Schritt 4: 12 15 21 22 [24] [31] [40] 48 55 59 71 88 91 96 97

Aufgabe 4

Fall 1: Für aufsteigend sortierte Daten:
InsertionSort, BubbleSort

In der Aufgabestellung steht, dass die Daten am schnellsten aufsteigend sortiert werden sollen. Im Fall 1 sind die Daten schon aufsteigend sortiert, also im besten Fall. Deswegen haben InsertionSort und BubbleSort die kleinste Anzahl der Vergleiche, und zwar jeweils circa n Vergleiche.

Fall 2: Für absteigend sortierte Daten:
MergeSort

Im Fall 2 sind die Daten absteigend sortiert und gerade umgekehrt zur Anforderung der Aufgabe. In diesem Fall hat MergeSort die kleinste Anzahl der Vergleiche $n \log_2 n$.

Fall 3: Für unsortierte Daten:
MergeSort, Quicksort
Für unsortierte Daten haben MergeSort und Quicksort loglineare Komplexität, und zwar $n \log_2 n$ bei MergeSort und $1,386 n \log_2 n$ bei QuickSort.