Siyu Chen (3494095) st169719@stud.uni-stuttgart.de
Xuefeng Hou (3502673) st175367@stud.uni-stuttgart.de
Leqi Xu (3556962) st176119@stud.uni-stuttgart.de

Aufgabe 1: Suchverfahren

(a) sequenzielle Suche

a) Folge 1 ohne Optimierung, gesuchtes Element:11
$index = 0, 1 < 11$
$index = 1, 23 > 11$
$index = 2, 47 > 11$
$index = 3, 43 > 11$
$index = 4, 68 > 11$
$index = 5, 12 > 11$
$index = 6, 11 = 11$ return index = 6

b) Folge 1 mit Optimierung, gesuchtes Element:11
$index = 0, 1 < 11$
$index = 1, 23 > 11$ return no_key

c) Folge 2, gesuchtes Element:48
$index = 0, 12 < 48$
$index = 1, 13 < 48$
$index = 2, 23 < 48$
$index = 3, 47 < 48$
$index = 4, 48 = 48$ return index = 4

(b) binäre Suche

a) Folge 1, gesuchtes Element:11
$m = (0 + 14)/2 = 7, F_1[m] = 73 > 11$
$m = (0 + 6)/2 = 3, F_1[m] = 43 > 11$
$m = (0 + 2)/2 = 1, F_1[m] = 23 > 11$
$m = 0, 1 < 11$ return no_key

b) Folge 2, gesuchtes Element:48
$m = (0 + 14)/2 = 7, F_2[m] = 73 > 48$
$m = (0 + 6)/2 = 3, F_2[m] = 47 < 48$
$m = (4 + 6)/2 = 5, F_2[m] = 68 > 48$
$m = 4, F_2[m] = 48 = 48$ return index = 4

Aufgabe 2: verkettete-Listen

Java-Programmierungsaufgabe

Aufgabe 3: BubbleSort

(a) die Liste nach den einzelnen Durchlauf

- 4, 8, 22, 2, 18, 32, 91, 50, 53, 67
- 4, 8, 2, 22, 18, 32, 91, 50, 53, 67

- 4, 8, 2, 18, 22, 32, 91, 50, 53, 67
- 4, 8, 2, 18, 22, 32, 50, 91, 53, 67
- 4, 8, 2, 18, 22, 32, 50, 53, 91, 67
- 4, 8, 2, 18, 22, 32, 50, 53, 67, 91
- 4, 2, 8, 18, 22, 32, 50, 53, 67, 91
- 2, 4, 8, 18, 22, 32, 50 ,53, 67, 91

(b) Für diese einfach verkettete List ist es unmöglich, die Knoten von hinten nach vorne zu iterieren.

Aufgabe 4: Sortierverfahren Komplexität

(a) MergeSort
Da die Folge absteigend sortiert (schlechtester Fall) ist, setzen sich MergeSort durch.
Mit O-Notation $O(nlog_2(n))$.

(b) BubbleSort und InsertionSort
Da die Folge F aufsteigend ist, es wird nur $n-1$ Vergleiche gebraucht.
Mit O-Notation $O(n)$.

(c) MergeSort und QuickSort
Da die Folge chaostisch (durchschnittlicher Fall) ist, setzen sich MergeSort und QuickSort durch. Da die Aufwandanalyse des MergeSort nlog2(n) ist
Mit O-Notation $O(nlog(n))$.