Aufgabe 4

Gegeben ist ein Array a von ganzen Zahlen der Länge n, z. B. :

i	0	1	2	3	4	5	6	7	8	9
a_i	5	-6	4	2	-5	7	-2	-7	3	5

Im Beispiel ist also $n = 10$. Es soll die maximale Teilsumme berechnet werden, also der Wert des Ausdrucks

$$\max_{i,j \leq n} \sum_{k=1}^{j-1} a_k$$

Im Beispiel ist dieser Wert 8 und wird für $i = 8, j = 10$ erreicht. Entwerfen Sie ein Divide-And-Conquer Verfahren, welches diese Aufgabenstellung in Zeit $\mathcal{O}(n \log n)$ löst. Skizzieren Sie Ihre Lösung hinreichend detailliert.

Tipp: Sie sollten ein geringfügig allgemeineres Problem lösen, welches neben der maximalen Teilsumme auch noch die beiden „maximalen Randsummen" berechnet. Die werden dann bei der Endausgabe verworfen.

```
3  /**
4   * Klasse zur Berechnung der maximalen Teilsumme einer Zahlenfolge.
5   *
6   * nach Teilsumme.java Klasse mit Algorithmen für die Berechnung des größten
7   * gemeinsamen Teilers zweier Ganzzahlen Algorithmen und Datenstrukturen,
8   * Auflage 4, Kapitel 2.1
9   *
10  * nach Prof. Grude, Prof. Solymosi, (c) 2000-2008: 22. April 2008
11  * http://public.beuth-hochschule.de/oo-plug/A&D/prog/kap21/Teilsumme.java
12  */
13 public class Teilsumme {
14
15   /**
16    * Berechne die maximale Teilsumme an der rechten Grenze. Die Eingabeparameter
17    * müssen diese Werte aufweisen: 0 <= links <= rechts < folge.length.
18    *
19    * @param folge  Die Zahlenfolge, in der die maximale Zahlensumme gerechnet
20    *               werden soll.
21    * @param links  Die Index-Nummer der linken Grenze.
22    * @param rechts Die Index-Nummer der rechten Grenze.
23    *
24    * @return Die maximale Teilsumme.
25    */
26   private static int berechneRandRechts(int[] folge, int links, int rechts) {
27     int max = 0;
28     int sum = 0;
29     for (int i = rechts; i >= links; i--) {
30       sum += folge[i];
31       max = Math.max(max, sum);
32     }
33     return max;
34   }
35
36   /**
37    * Berechne die maximale Teilsumme an der linken Grenze. Die Eingabeparameter
38    * müssen diese Werte aufweisen: 0 <= links <= rechts < folge.length.
39    *
40    * @param folge  Die Zahlenfolge, in der die maximale Zahlensumme gerechnet
```

```
41      *               werden soll.
42      * @param links  Die Index-Nummer der linken Grenze.
43      * @param rechts Die Index-Nummer der rechten Grenze.
44      *
45      * @return Die maximale Teilsumme.
46      */
47    private static int berechneRandLinks(int[] folge, int links, int rechts) {
48      int max = 0;
49      int sum = 0;
50      for (int i = links; i <= rechts; i++) {
51        sum += folge[i];
52        max = Math.max(max, sum);
53      }
54      return max;
55    }
56
57    /**
58     * Berechne die maximale Teilsumme in der Zahlenfolge zwischen einer gegeben
59     * linken und rechten Grenze. Die Eingabeparameter müssen diese Werte
      ↪  aufweisen:
60     * 0 <= links <= rechts < folge.length.
61     *
62     * @param folge  Die Zahlenfolge, in der die maximale Zahlensumme gerechnet
63     *               werden soll.
64     * @param links  Die Index-Nummer der linken Grenze.
65     * @param rechts Die Index-Nummer der rechten Grenze.
66     *
67     * @return Die maximale Teilsumme.
68     */
69    private static int berechne(int[] folge, int links, int rechts) {
70      if (links == rechts) // nur ein Element
71        return Math.max(0, folge[links]);
72      else {
73        final int mitte = (rechts + links) / 2;
74        final int maxLinks = berechne(folge, links, mitte);
75        final int maxRechts = berechne(folge, mitte + 1, rechts);
76        final int maxGrenzeRechts = berechneRandRechts(folge, links, mitte);
77        // linke Hälfte
78        final int maxGrenzeLinks = berechneRandLinks(folge, mitte + 1, rechts);
79        // rechte Hälfte
80        return Math.max(maxRechts, Math.max(maxLinks, maxGrenzeRechts +
          ↪  maxGrenzeLinks));
81      }
82    }
83
84    /**
85     * Berechne die maximale Teilsumme einer Zahlenfolge rekursiv mit
86     * logarithmischer Zeitkomplexität.
87     *
88     * @param folge Die Zahlenfolge, in der die maximale Zahlensumme gerechnet
89     *              werden soll.
90     *
91     * @return Die maximale Teilsumme.
92     */
93    public static int berechne(int[] folge) {
94      return berechne(folge, 0, folge.length - 1);
95    }
96
97    public static void main(String[] args) {
98      int[] folge = { 5, -6, 4, 2, -5, 7, -2, -7, 3, 5 };
99      int ergebnis = berechne(folge);
100     System.out.println(ergebnis);
```

```
101         }
102     }
```

Code-Beispiel auf Github ansehen: `src/main/java/org/bschlangaul/examen/examen_66115/jahr_2012/herbst/Teilsumme.java`