CHRISTIAN-ALBRECHTS-UNIVERSITÄT ZU KIEL
Institut für Informatik, Arbeitsgruppe Theorie der Parallelität
Prof. Dr. K. Jansen, K.-M. Klein, M.Maack

21. Mai 2015

Hausaufgaben zur Vorlesung »Algorithmen und Datenstrukturen«

Blatt 6

Hausaufgabe 6.1 (Mergesort (2 Punkte))
Der Speicheraufwand des naiven Mergesort-Algorithmus lässt sich angeben durch eine Funktion $M : \mathbb{N}_{>0} \mapsto \mathbb{N}$, die definiert ist durch

$$M(n) = \begin{cases} 0 & \text{falls } n = 1 \\ n + M(\lceil \frac{n}{2} \rceil) & \text{sonst} \end{cases}$$

Zeigen Sie für alle $n \in \mathbb{N}_{>0}$ die folgenden Ungleichungen:

$$2(n-1) \leq M(n) \leq 2(n-1) + \lceil \log_2(n) \rceil$$

Hinweis: Für alle $n \in \mathbb{N}$ gilt, dass $\lceil \log_2(\lceil \frac{n}{2} \rceil) \rceil = \lceil \log_2(n) \rceil - 1$.

Hausaufgabe 6.2 (Max- und Min-Suche (2 Punkte))
Entwickeln Sie einen Algorithmus, der zu einem ganzzahligen Feld das Maximum und das Minimum der Feldelemente bestimmt. Bemühen Sie sich, für ein Feld der Länge n mit $3/2n + O(1)$ Vergleichen von Feldelementen auszukommen.

Hausaufgabe 6.3 (Programmierung (2 Punkte))
Implementieren Sie eine doppelt verkettete Liste in Java. Vervollständigen Sie dafür die folgende Klasse List (siehe auch Vorlage):

```
class Node {
  int value;
  Node prev;
  Node next;
}
class List {
  Node first;
  Node last;
  int size;

// inserts element after index x
   static void insert(List l, int index, int element) {
   }

// removes element with index index and returns its value
   static int remove(List l, int index) {
   }

// returns value of element with index index
   static int get(List l, int index) {
   }
```

```
// returns the size of the list
   static int size(List l) {
   }

// initializes an empty list
   static void init(List l){
   }
}
```

Verwenden Sie die von uns bereitgestellten Dateien und testen Sie Ihr Programm mit dem bereitgestellten Testprogramm. Verwenden Sie zur Ausführung des Testprogramms den Befehl "python test.py List <Pfad>". Abgaben, die sich nicht mit dem Testprogramm testen lassen, werden mit 0 Punkten bewertet.

Hinweis: Die Liste lässte sich einfacher handhaben wenn sich am Anfang und am Ende der Liste zwei Dummy-Elemente befinden.

Hausaufgabe 6.4 (Backtracking (4 Punkte))
Wir betrachten beispielsweise folgendes Sodoku.

	2		1			4		
4								
1	8		4	5				6
6	5		8		4			2
		3				1		
					3		6	9
9			3		6		5	4
								3
5		2			1		8	

Bei diesem Zahlenrätsel sollen Zahlen aus $\{1,\ldots,9\}$ in die noch freien Felder so eingetragen werden, dass die folgenden Bedingungen erfüllt sind.

i) In jeder der neun Spalten und Zeilen muss jede Zahl aus $\{1,\ldots,9\}$ genau einmal vorkommen.

ii) In jedem der abgesetzten 3×3-Blöcke muss jede Zahl aus $\{1,\ldots,9\}$ genau einmal vorkommen.

Das Sudoku ist dabei gegeben als ein zwei dimensionales Array *sudoku* mit Zahlen aus $\{0,\ldots,9\}$. Hierbei steht $sudoku[i][j] = 0$ für ein freies Feld an Position (i, j) für $i, j \in \{0,\ldots,8\}$. Geben Sie ein Programm in Java an, welches das Problem mittels Backtracking löst.

Verwenden Sie die von uns bereitgestellten Dateien und testen Sie Ihr Programm mit dem bereitgestellten Testprogramm. Verwenden Sie zur Ausführung des Testprogramms den Befehl "python test.py Sudoku <Pfad>". Abgaben, die sich nicht mit dem Testprogramm testen lassen, werden mit 0 Punkten bewertet.

Abgabe der theoretischen Aufgaben Donnerstag, den 28. Mai, bis spätestens 11 Uhr im Schrein. Die Abgabe der Programmieraufgaben erfolgt per EMail an Ihren Übungsleiter.