CHRISTIAN-ALBRECHTS-UNIVERSITÄT ZU KIEL
Institut für Informatik, Arbeitsgruppe Algorithmen und Komplexität
Prof. Dr. K. Jansen, K.-M. Klein, F. Land M. Rau

19. Mai 2016

Hausaufgaben zur Vorlesung »Algorithmen und Datenstrukturen«

Blatt 6

Hausaufgabe 6.1 (Stack (3 Punkte))
In der folgenden Sequenz steht ein Buchstabe für eine *push*-Operation und ein Sternchen für eine *pop*-Operation eines Stacks:

`E A S * Y * Q U E * * * S T * * * I O * N * * *`

(a) Geben Sie die Sequenz der Werte an, die die *pop*-Operationen zurückgeben, wenn diese Sequenz von Operationen auf einem anfänglich leeren Stack ausgeführt wird.

(b) Modifizieren Sie die Sequenz durch Verschieben der Sternchen derart, dass folgende Ausdrücke zurückgeliefert werden, oder belegen Sie durch Argumente, dass es für die einzelnen Fälle keine derartige Sequenz gibt.

(1) `E A S Y Q U E S T I O N`

(2) `N O I T S E U Q Y S A E`

(3) `U E Q S Y T S I A O E N`

Hausaufgabe 6.2 (Binärbäume (3 Punkte))
In einem Baum nennt man die Blätter auch externe Knoten. Interne Knoten sind dementsprechend genau die Knoten, die keine externen Knoten sind. Eine Kante oder Verbindung besteht zwischen zwei Knoten, wenn die beiden Knoten in "Vater-Kind-Beziehung" stehen.
Ein echter binärer Baum ist ein Baum mit der Eigenschaft, dass jeder interne Knoten genau zwei Kinder hat.

Beweisen Sie die folgenden beiden Sätze für nichtleere echte binäre Bäume:

(a) Ein nichtleerer echter Binärbaum mit N internen Knoten hat $N+1$ externe Knoten.
Hinweis: Nutzen Sie aus, dass die Induktionsvoraussetzung insbesondere für echte Teilbäume gilt und (induzierte) Teilbäume von binären Bäumen selbst wieder binär sind.

(b) Ein echter Binärbaum mit N internen Knoten hat $2N$ Kanten.

Hausaufgabe 6.3 (Binärbäume (4 Punkte))
Beweisen Sie die folgenden Behauptungen für vollständige Binärbäume. Für den Knoten mit der Nummer $i \in \{0, \ldots, n-1\}$ gilt:

$$\text{left}(i) = 2i + 1 \tag{1}$$

$$\text{right}(i) = 2i + 2 \tag{2}$$

$$\text{parent}(i) = \lfloor \frac{i-1}{2} \rfloor, \tag{3}$$

wobei left(i) die Nummer des linken und right(i) die Nummer des rechten Nachfolgers von i ist und parent(i) die Nummer des Vorgängers von i, falls diese Knoten existieren.

Hinweis: Beweisen Sie die Behauptung zunächst für alle "links aussen" liegenden Knoten.

Programmieraufgabe 6.4
Implementieren Sie einen Algorithmus basierend auf der Backtracking Methode, um einen Weg vom Start zum Ziel in einem Labyrinth zu finden, oder um zu entscheiden, dass kein entsprechender Weg existiert. Das Labyrinth ist durch einen ungerichteten Graphen $G = (V, E)$ modelliert. Der Start und das Ende des Labyrinths sind durch zwei ausgezeichnete Knoten s, t in dem Graphen gegeben. Gesucht ist ein Pfad in dem Graphen vom Startknoten s zum Zielknoten t.

Abgabe der theoretischen und praktischen Aufgaben Donnerstag, den 26. Mai, bis spätestens 14 Uhr. Die Abgabe der theoretischen Aufgaben erfolgt im Schrein. Die Abgabe der praktischen Aufgaben erfolgt im iLearn.