Algorithmen und
Berechnungskomplexität I
Wintersemester 2017/18

Abgabe: 21.11.2017, 12.00 Uhr
Besprechung: KW 48

UNIVERSITÄT BONN

PD Dr. Elmar Langetepe
David Kübel
Institut für Informatik

Übungsblatt 6

Aufgabe 6.1: (2+4 Punkte)

Betrachten Sie die in der Vorlesung vorgestellten Datenstrukturen Stack (auch Stapel oder Keller) und dynamische Liste:

(a) Wie kann ein Stack durch eine einfach verkettete Liste realisiert werden, so dass die Operationen PUSH und POP konstante Zeit ($O(1)$) benötigen?

(b) Beschreiben Sie ein Verfahren, um mit Hilfe *nur eines* n-elementigen Arrays $A[1, \ldots, n]$ *zwei* Keller zu implementieren. Die Fehlermeldung "Keller voll" soll erst dann erscheinen, wenn die Gesamtzahl der Elemente in beiden Kellern zusammen den Wert n übersteigt. Dabei sollen die Operationen PUSH($keller$, $element$) und POP($keller$) mit $keller \in \{1, 2\}$ in konstanter Zeit laufen.

Aufgabe 6.2: (4 Punkte)

Sei T ein binärer Suchbaum und v ein innerer Knoten in T, der keine zwei Blätter als Kinder hat. Zeigen Sie, dass der Knoten w, der den bezüglich v nächstgrößeren Schlüssel enthält, mindestens ein Blatt als Kind hat.

Aufgabe 6.3: (4+4 Punkte)

Überlegen Sie sich, wie man einen Binärbaum T in einem Array implementieren kann, sodass damit Wurzel(T), links(v), rechts(v) und parent(v) für jedes $v \in T$ in konstanter Zeit bestimmt werden kann.

Fügen Sie in einen leeren binären Suchbaum nacheinander die Werte 4, 1, 3, 2, 16 und 9 ein, sodass die Suchbaumeigenschaft aufrecht erhalten bleibt. Löschen Sie anschließend die Knoten mit den Schlüsseln 2 und 4. Geben Sie für jeden Schritt die Darstellung als Graphen, sowie den Inhalt des Arrays an!

Aufgabe 6.4: (6 Punkte)

Betrachten wir die Menge der Zeichenketten über dem Alphabet $\{(,)\}$, also alle Zeichenketten, die nur aus den beiden Klammerzeichen (und) bestehen. Eine solche Zeichenkette z der Länge $2n$ heißt *zulässige Klammerung*, falls z aus genau n öffnenden und n schließenden Klammern besteht, und gleichzeitig jeder Präfix (Anfangsabschnitt) von z mindestens so viele öffnende wie schließende Klammersymbole enthält. Diese Definition entspricht der intuitiven; so ist z.B. `()` zulässig geklammert, nicht jedoch `)(` – hier stimmt zwar die Anzahl der beiden Klammerarten, jedoch enthält der Präfix `)` mehr schließende als öffnende Klammern. Analog ist `()(()((()))())))` zulässig geklammert, aber `())(()` nicht. Betrachten wir weiter die Menge der *vollen Binärbäume*. Ein voller Binärbaum ist ein Binärbaum, bei dem jeder innere Knoten genau zwei Kinder hat.

Nun zur Aufgabe. Zeigen Sie, dass folgende zwei Mengen gleichmächtig sind:

1. Die Menge aller zulässigen Klammerungen der Länge $2n$,
2. die Menge aller vollen Binärbäume mit n inneren Knoten

Gleichmächtigkeit von Mengen kann man zeigen, indem man eine Bijektion zwischen den beiden Mengen angibt. Eine solche Bijektion kann beispielsweise durch einen Algorithmus beschrieben werden, der jedem Klammerausdruck eineindeutig einen Binärbaum zuordnet, oder umgekehrt.

Aufgabe 6.5: (4 Zusatzpunkte)

Sei T ein Binärbaum mit n Knoten. Für einen Knoten v bezeichnen $\text{pre}(v)$ und $\text{post}(v)$ die Position (also bspw. $1, 2, 3 \dots$) von v in der Besuchsreihenfolge eines Prä- bzw. Postorderdurchlauf von T.

Zeigen Sie, dass für beliebige Knoten $a, b \in T$ gilt:

$$a \text{ Vorgänger von } b \iff \text{pre}(a) < \text{pre}(b) \ \wedge \ \text{post}(a) > \text{post}(b).$$

Entwerfen Sie einen Algorithmus, welcher nach $O(n)$ Vorbereitungszeit Anfragen des Typs *"Ist a Vorgänger von b?"* in konstanter Zeit beantworten kann. Wann bietet Ihr Algorithmus einen Vorteil im Vergleich zum naiven Algorithmus, welcher bei jedem Aufruf alle Vorgänger von b nach a durchsucht?