Datenstrukturen und Algorithmen
Übung 7, Frühling 2018

12. April 2018

Abgabe: Diese Übung muss zu Beginn der Übungsstunde bis spätestens um 16 Uhr 15 am 19. April abgegeben werden. Die Abgabe der DA Übungen erfolgt immer in schriftlicher Form auf Papier. Programme müssen zusammen mit der von Ihnen erzeugten Ausgabe abgegeben werden. Drucken Sie wenn möglich platzsparend 2 Seiten auf eine A4-Seite aus. Falls Sie mehrere Blätter abgeben heften Sie diese bitte zusammen (Büroklammer, Bostitch, Mäppchen). *Der gesammte Sourcecode muss ausserdem elektronisch über Ilias abgegeben werden.*

Die Übung sollte vorzugsweise in Zweiergruppen bearbeitet werden, kann aber auch einzeln abgegeben werden. Vergessen Sie nicht, Ihren Namen und Ihre Matrikelnummer auf Ihrer Abgabe zu vermerken. Jede Übungsserie gibt 10 Punkte. Im Durchschnitt müssen Sie 7 von 10 Punkten erreichen, um die Testatbedingungen zu erfüllen.

Theoretische Aufgaben: Binäre Suchbäume

1. Gegeben seien die Schlüssel $\{2, 4, 9, 13, 17, 21, 24\}$. Zeichnen Sie binäre Suchbäume der Höhe $2, 3, 4, 5$ und 6. **1 Punkt**

2. Was ist der Unterschied zwischen der binären Suchbaum-Eigenschaft und der Min-Heap Eigenschaft? Kann die Min-Heap Eigenschaft benutzt werden, um Schlüssel in einem Baum mit n Knoten in sortierter Reihenfolge in $O(n)$ Zeit auszugeben? Gibt es einen vergleichenden Algorithmus, der für beliebige Schlüsselfolgen einen binären Suchbaum in $O(n)$ aufbaut? Erklären Sie Ihre Antwort. **1 Punkt**

3. Angenommen, die Suche nach einem Schlüssel k in einem binären Suchbaum endet in einem Blatt. Wir unterscheiden drei Mengen: A, die Schlüssel links vom Suchpfad, B die Schlüssel auf dem Suchpfad, und C, die Schlüssel rechts vom Suchpfad. Die Vermutung ist, dass für jeweils drei Schlüssel $a \in A, b \in B$ und $c \in C$ gilt, dass $a \leq b \leq c$. Widerlegen Sie diese Vermutung mit einem Gegenbeispiel, das einen möglichst kleinen Baum verwendet. **1 Punkt**

4. Schreiben Sie Pseudocode für eine rekursive Version des Einfügens eines Knotens in einen nicht leeren binären Suchbaum. Beschreiben Sie ihren Algorithmus in 1-2 Sätzen. **1 Punkt**

5. Ein Knoten x in einem binären Suchbaum habe zwei Kinder. Zeigen Sie, dass der Nachfolger von x kein linkes Kind und der Vorgänger von x kein rechtes Kind hat. **1 Punkt**

Theoretische Aufgaben: Repetition

In diesem Teil werden prüfungsrelevante Aufgabenstellungen aus den vorderen Kapiteln wiederholt.

1. Gegeben seien zwei *zyklische, doppelt verkettete* Listen a und b. Nehmen Sie an, die Listen hätten je ein Wächterelement $nil[a]$ und $nil[b]$. Geben Sie Pseudocode für eine Funktion $concatenate(nil[a], nil[b])$, welche der Liste a alle Elemente der Liste b anhängt. Nach Aufruf von *concatenate* besteht also die Liste a aus den Elementen von a gefolgt von den Elementen von b. Das Wächterelement $nil[b]$ soll natürlich nicht in a eingefügt werden. Behandeln Sie die Fälle korrekt, wo a und/oder b keine Elemente ausser dem Wächter enthalten. Ihr Algorithmus soll eine Zeitkomplexität von $O(1)$ haben. **1 Punkt**

2. Ordnen Sie die folgenden Funktionen nach ihrer asymptotischen Wachstumsrate:

   (a) $\sqrt{n^7}$
   (b) 2^n
   (c) $\frac{n!}{n^n}$
   (d) $\log(\log(4n))$
   (e) $3n^2 + 2n + 1$
   (f) $5n - 2$
   (g) $2^{log_3(n)}$
   (h) $n^3 \log(n)$
   (i) 2^{12^8}

   1 Punkt

3. Geben Sie die asymptotische Laufzeit in Abhängigkeit von n für folgenden Algorithmus in Theta-Notation an:

```
1  i ← 1
2  j ← 1
3  while i <= n
4        do
5           while j <= i + 5
6                 do
7                       j ← j + 1
8           i ← i + 1
```

   1 Punkt

4. Gegeben ist die folgende Rekursionsgleichung:

$$T(n) = \begin{cases} 9T(\frac{n}{3}) - 1 & \text{wenn } n > 1 \\ \frac{1}{4} & \text{wenn } n = 1 \end{cases}$$

   Beweisen Sie mit vollständiger Induktion, dass die Rekursionsgleichung die Lösung $\frac{n^2+1}{8}$ hat. **1 Punkt**

5. In einem Naturgarten möchte man einen geraden Weg durch eine grosse Wiese bauen. Der Weg muss vom einen Ende zum anderen gehen, muss also eine exakt festgelegte Gesamtlänge L (ganzzahlig, in cm) haben. Er soll aus Granitplatten gelegt werden, von denen eine grosse Menge M zum Kauf bereitsteht. Alle verfügbaren Granitplatten haben die gleiche Breite, gerade so wie für den Weg gewünscht, haben aber ganz unterschiedliche Einzellängen l_i (ganzzahlig, in cm). Jede Granitplatte hat ihren Preis p_i, unabhängig von ihrer Einzellänge, aufgrund des unterschiedlichen Herstellers. Wir möchten nun einige der Granitplatten kaufen, so dass exakt die gewünschte Gesamtlänge des Wegs getroffen wird, dass wir aber insgesamt für die gekauften Platten möglichst wenig bezahlen müssen.

   Beispiel: Unser Weg ist 100 cm lang, und es stehen Platten mit (Länge, Preis) von $(60, 5)$, $(55, 6)$, $(35, 7.5)$, $(25, 12)$, $(15, 14)$, und $(40, 32)$ zur Verfügung. Wir kaufen die erste, vierte und fünfte Platte in dieser Aufzählung.

   Entwerfen Sie ein rekursives Programm in Pseudocode, welches den minimalen Preis für eine genau passende Auswahl von Platten berechnet (und Null liefert, falls es keine solche Auswahl gibt). Geben Sie die Laufzeit Ihres Algorithmus an. **1 Punkt**