Übungen zur Vorlesung Einführung in die Informatik II

Ausgabe: 4. Juni 2002 Abgabe: 10. Juni 2002, 12.00 Uhr

Die Abgabe erfolgt ausschließlich über das System eClaus! Siehe
http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ss02/info_II_02.html

Aufgabe 1: (Bäume) **schriftlich**

(5 Punkte) Schreiben Sie ein Programm, das binäre Zufallsbäume mit Hilfe des `tree` Paketes vom Aufgabenblatt 6 erzeugt. Dazu soll eine Zahl p mit $0 \leq p \leq 0.5$ eingegeben werden, die angibt, mit welcher Wahrscheinlichkeit ein linker Unterbaum erzeugt wird (dieser wird dann wieder mit gleicher Wahrscheinlichkeit Unterbäume bekommen), der rechte Unterbaum soll unabhängig vom linken ebenfalls mit Wahrscheinlichkeit p erzeugt werden. Alle Knoten bleiben unbeschriftet.
Welcher Traversierungsart entspricht die Reihenfolge, in der Sie die Knoten in dem Baum erzeugt haben?
Untersuchen Sie die Höhe der so erzeugten Bäume für $p = 0.05, 0.10, 0.15, \ldots, 0.4$ im Mittel von jeweils 100 zufällig erzeugten Bäumen. Was ist bei Werten $p > 0.5$ zu erwarten?

Aufgabe 2: (Eigenschaften von Bäumen) **schriftlich**

(3 Punkte) Schreiben Sie ein Programm, das für binäre Zufallsbäume (z.B. aus Aufgabe 1) jeden Knoten v mit der Anzahl der Knoten im Teilbaum mit der Wurzel v beschriftet. Ein leerer Unterbaum hat dabei 0 Knoten, die Anzahl der Knoten im Teilbaum ist die Anzahl der Knoten in den beiden Unterbäumen plus den Knoten v.

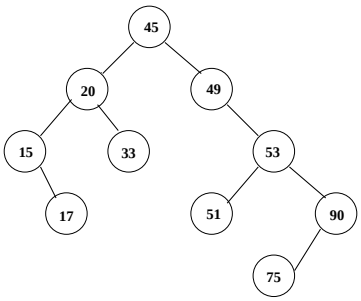
Aufgabe 3: (Baumalgorithmen) **zum Votieren**

Betrachten Sie folgenden Algorithmus (die Typ-Bezeichnungen sind an Kapitel 3.7 angelehnt):

```
procedure wastutdas (v : NodePtr) is
   r : NodePtr := new Node;
   p : NodePtr := r;
   c : NodePtr := v;
   h : NodePtr;
begin
   r.left:=v;
   while c /= r loop
      if c /= null then
         Put (c.node);
         h := c.left;
         c.left := c.right;
         c.right := p;
         p := c;
         c := h;
      else
         h := c;
         c := p;
         p := h;
      end if;
   end loop;
end wastutdas;
```

(5 Punkte) Was tut diese Prozedur (wie wird der Unterbaum von v verändert) und welche Ausgabe liefert sie (vergleichen Sie die Ausgabe mit anderen Ihnen bereits bekannten Baumalgorithmen)?

Gegeben sei der hier abgebildete binäre Suchbaum.

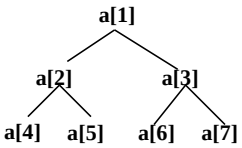


a. (2 Punkte) Geben Sie eine mögliche Reihenfolge an, in der die Elemente in einen anfangs leeren Suchbaum eingefügt worden sein könnten, damit genau dieser Suchbaum entsteht. In welcher Reihenfolge hätten sie eingefügt werden müssen, damit sich ein vollständig ausgeglichener Suchbaum ergibt (es sind mehrere Reihenfolgen möglich, geben Sie nur *ein* Beispiel an)? Ist es demnach praktisch möglich, für jeden beliebigen gegebenen Suchbaum einen gleichwertigen ausgeglichenen Suchbaum zu erzeugen? (Skizzieren Sie einen Algorithmus oder geben Sie ein Gegenbeispiel an)

b. (2 Punkte) Zeichnen Sie den Baum, wie er nach den unten aufgeführten Operationen aussieht. Für den Fall, dass ein Knoten mit zwei nicht-leeren Teilbäumen gelöscht werden soll, sind im Skript zwei (symmetrische) Lösungen angegeben. Zeichnen Sie für diesen Fall beide auf. (Hinweis: Es handelt sich nicht um balancierte Suchbäume, d.h. Rotationen sind *nicht* erforderlich.)

- Löschen von 53
- Einfügen von 88
- Löschen von 49
- Einfügen von 49

Aufgabe 5: (Suchbäume implementiert durch Arrays) **zum Votieren**



In der Vorlesung haben Sie die Implementierung eines binären Suchbaumes mit Hilfe von Zeigern kennengelernt. Sie können Suchbäume aber auch als ARRAY realisieren, indem Sie den einzelnen Knoten des Baumes bestimmte Indizes des ARRAYs zuordnen. Wenn ein Knoten im ARRAY unter Index i gespeichert ist, beträgt der Index des linken Sohnknotens $2 \cdot i$ und der des rechten Sohnes $2 \cdot i + 1$. Die Wurzel wird unter Index 1 gespeichert. Der Baum wird also schichtenweise nacheinander im ARRAY abgelegt.
Das Einfügen soll nach dem in der Vorlesung kennengelernten Verfahren geschehen: Zunächst wird in dem Baum so lange gesucht, bis das Element entweder gefunden wird (Element bereits vorhanden, kein weiteres Einfügen) oder bis ein leerer Knoten gefunden wird, in dem das neue Element eingefügt wird.

a. (1.5 Punkte) In die oben beschriebene (zu Beginn leere) Struktur sollen nacheinander folgende natürliche Zahlen eingefügt werden: 16, 10, 5, 20, 26, 1, 7, 25, 27. Geben Sie den Baum in seiner ARRAY-Darstellung an.

b. (1.5 Punkte) Wie gross müssen Sie das ARRAY dimensionieren, um neun beliebige Elemente hintereinander in den Suchbaum einfügen zu können? Ist diese Art der Baumdarstellung für beliebige Elementmengen sinnvoll? Begründen Sie Ihre Antworten.