Universität Stuttgart
Fakultät 5, Institut FMI
Einführung in die Informatik II

Sommersemester 2006
Claus, Draskoczy, Schmid
12.05.06

Aufgabenblatt 4 **Abgabe: 20.05.06, 24.00 Uhr**

1. **Preorder** (mittel) (schriftlich, 5 Punkte)
Durch die Preorder-Ausgabe eines Suchbaums lässt sich dieser Suchbaum wieder eindeutig rekonstruieren.

   a) Schreiben Sie ein Ada-Programm, das eine Folge ganzer Zahlen vom Benutzer einliest und daraus den dazugehörigen Suchbaum rekonstruiert. Der erzeugte Suchbaum soll auf dem Bildschirm wieder ausgegeben werden.

   b) Gibt es zu jeder beliebigen Folge von ganzen Zahlen, die als Preorder-Ausgabe interpretiert wird, einen Suchbaum? Falls nicht, soll Ihr Programm in diesem Fall eine klare Fehlermeldung ausgeben!

2. **Postorder** (mittel–schwer) (schriftlich, 2 Punkte)
Durch die Postorder-Ausgabe eines Suchbaums lässt sich dieser Suchbaum wieder eindeutig rekonstruieren.

   a) Überlegen Sie sich einen Algorithmus und geben Sie eine ausführliche Beschreibung der Idee im eClaus-System ab. Ist diese Aufgabe schwieriger als bei der Preorder-Ausgabe?

   b) Wer einen korrekten Algorithmus implementiert, erhält 7 Bonuspunkte!

3. **Löschwahrscheinlichkeit eines Blattes** (mittel) (schriftlich, 3 Punkte)
In einem Binärbaum ist das Löschen eines Blattes besonders einfach. Die durchschnittliche Wahrscheinlichkeit (über alle C_n-Bäume gemittelt), dass ein Blatt gelöscht wird, ist:

$$P_n^{\mathrm{Bl}} = \frac{1}{C_n} \sum_{i=1}^{C_n} \frac{\text{Blätter im Baum } B_i}{n}$$

   a) Berechnen Sie P_3^{Bl}, P_4^{Bl} und P_5^{Bl} von Hand.

   b) Falls der Baum zu einer Liste entartet ist, dann ist die Löschwahrscheinlichkeit $\frac{1}{n}$ (es gibt nur ein Blatt im Baum). Wieviele solche entarteten Bäume mit n Knoten gibt es?

4. **Ein weiterer Baumalgorithmus** (mittel) (schriftlich, 5 Punkte)
Betrachten Sie folgenden Algorithmus (`NodePtr` ist ein `access` Typ auf einen (Binär-) Baumknoten `Node`):

```
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.value);
        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;
```

a) 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)?

b) Welche Laufzeit in O-Notation hat der Algorithmus, wenn der Baum n Knoten hat?

5. Optimale Suchbäume I (leicht) (schriftlich, 3 Punkte)

Bei optimalen Suchbäumen versucht man die Knoten im Suchbaum so anzuordnen, dass die mittlere Suchdauer für Elemente des Baumes minimiert wird. Es seien die Elemente 2, 3, 4, 5 mit den relativen Zugriffshäufigkeiten $p(2) = 0.3$, $p(3) = 0.1$, $p(4) = 0.5$, $p(5) = 0.1$ gegeben. Bestimmen Sie für alle möglichen Suchbäume mit diesen 4 Elementen die Preorder, die innere Pfadlänge und mittlere Suchdauer. Geben im eClaus für jeden Suchbaum diese Werte an, also z.B.

```
Preorder: 2 3 4 5, innere Pfadlänge: <..>, mittlere Suchdauer: <..>
```

6. Optimale Suchbäume II (mittel) (schriftlich, 2+4 Punkte)

Bei optimalen Suchbäumen versucht man die Knoten im Suchbaum so anzuordnen, dass die mittlere Suchdauer für Elemente des Baumes minimiert wird. Die relative Zugriffshäufigkeit p habe für die Zahlen 2, 5, 7, 8, 10 und 15 die Werte $p(2) = 0.1$, $p(5) = 0.2$, $p(7) = 0.2$, $p(8) = 0.1$, $p(10) = 0.3$ und $p(15) = 0.1$.

a) Bestimmen Sie den optimalen Suchbaum mit obigen Zahlen und Zugriffshäufigkeiten und dessen innere Pfadlänge und mittlere Suchdauer.

b) In der Praxis ist nicht immer vorab bekannt, ob ein Element auch im Baum vorkommt. Sind insgesamt Zahlen von 1 bis 20 erlaubt, so seien die relativen Häufigkeiten für die vorkommenen Zahlen $p(2) = 0.04$, $p(5) = 0.08$, $p(7) = 0.08$, $p(8) = 0.04$, $p(10) = 0.12$ und $p(15) = 0.04$ und für die „Lücken“ $p(1) = 0.04$, $p(3-4) = 0.08$, $p(6) = 0.04$, $p(9) = 0.04$, $p(11-14) = 0.16$, $p(16-20) = 0.24$. Bestimmen Sie für diese Situation den optimalen Suchbaum. Geben Sie im eClaus die Preorder, innere Pfadlänge und mittlere Suchdauer an. Wie groß ist die mittlere Suchdauer nun, wenn man sich auf die erfolgreichen Fälle beschränkt?