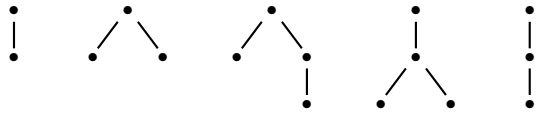
7. Übungsblatt zu Programmierung 1, WS 2008/09

Prof. Dr. Gert Smolka, Mark Kaminski, M.Sc.
www.ps.uni-sb.de/courses/prog-ws08/

Lesen Sie im Buch: Kapitel 7

Aufgabe 7.5 Geben Sie die Gestalt des Ausdrucks $(x + 3)(y + 7)$ an.

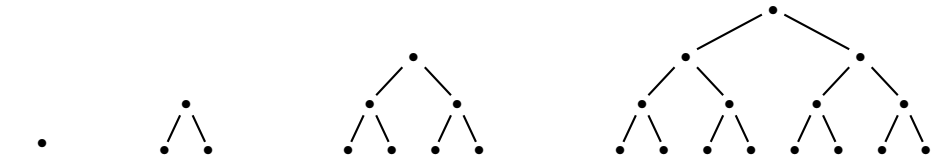
Aufgabe 7.7' Ordnen Sie die folgenden Bäume gemäß der lexikalischen Ordnung an:



Aufgabe 7.9 Geben Sie einen Baum mit 5 Knoten an, der genau zwei Teilbäume hat. Wie viele solche Bäume gibt es?

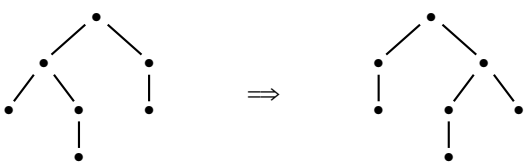
Aufgabe 7.10 Schreiben Sie eine Prozedur *binary* : *tree* → *bool*, die testet, ob ein Baum binär ist.

Aufgabe 7.11 Schreiben Sie eine Prozedur *tree* : *int* → *tree*, die für $n \geq 0$ binäre Bäume wie folgt liefert:



Achten Sie darauf, dass die identischen Unterbäume der zweistelligen Teilbäume jeweils nur einmal berechnet werden. Das sorgt dafür, dass Ihre Prozedur auch für $n = 1000$ schnell ein Ergebnis liefert. Verwenden Sie die Prozedur *iter* aus § 3.4.

Aufgabe 7.12 (Spiegeln) Spiegeln reversiert die Ordnung der Unterbäume der Teilbäume eines Baums:



Schreiben Sie eine Prozedur *mirror* : *tree* → *tree*, die Bäume spiegelt.

Aufgabe 7.14 Die arithmetischen Ausdrücke aus § 6.4 lassen sich wie in § 7.1.2 besprochen als Bäume auffassen. Schreiben Sie analog zu *ast* eine Prozedur *ase* : *exp* → *int list* → *exp*, die zu einem Ausdruck und einer Adresse den entsprechenden Teilausdruck liefert. Beispielsweise soll *ase* für den Ausdruck $x(2y + 3)$ und die Adresse $[2, 1]$ den Teilausdruck $2y$ liefern. Für ungültige Adressen soll die Ausnahme *Subscript* geworfen werden.

Aufgabe 7.15 (d) Schreiben Sie eine Prozedur *leaf* : *tree* → *int list* → *bool*, die testet, ob eine Adresse ein Blatt eines Baums bezeichnet.

Aufgabe 7.20 Schreiben Sie eine Prozedur *superior* : *int list* → *int list* → *bool*, die für zwei Adressen a und a' testet, ob es einen Baum gibt, in dem a einen Knoten bezeichnet, der dem durch a' bezeichneten Knoten übergeordnet ist.

Aufgabe 7.21 Die **Breite** eines Baums ist die Anzahl seiner Blätter. Beispielsweise hat der Baum $t3$ die Breite 7. Entwickeln Sie eine Prozedur *breadth* : *tree* → *int*, die die Breite eines Baums bestimmt.

Aufgabe 7.22 Der **Grad** eines Baums ist die maximale Stelligkeit seiner Teilbäume. Beispielsweise hat der Baum $t3$ den Grad 3. Entwickeln Sie eine Prozedur *degree* : *tree* → *int*, die den Grad eines Baums bestimmt.

Aufgabe 7.26 Deklarieren Sie mithilfe von *fold* eine Prozedur

a) *depth* : *tree* → *int*, die die Tiefe eines Baums bestimmt.
b) *breadth* : *tree* → *int*, die die Breite eines Baums bestimmt.
c) *degree* : *tree* → *int*, die den Grad eines Baums bestimmt.
d) *mirror* : *tree* → *tree*, die einen Baum spiegelt.

Aufgabe 7.30 Schreiben Sie eine Prozedur *prest'* : *tree* → *int* → *tree*, die zu einem Baum und einer Pränummer den entsprechenden Teilbaum liefert. Wenn die angegebene Zahl keine Pränummer des Baums ist, soll die Ausnahme *Subscript* geworfen werden.

Aufgabe 7.32 Schreiben Sie eine Prozedur *post'* : *tree* → *int* → *tree*, die zu einem Baum und einer Postnummer den entsprechenden Teilbaum liefert. Realisieren Sie die Agenda von *post* mit der folgenden Typdeklaration:

```
datatype 'a entry = I of 'a | F of 'a
```

Aufgabe 7.33 Geben Sie die Prä- und die Postlinearisierung des Baums $T[T[\,], T[T[\,]], T[T[\,], T[\,]]]$ an.

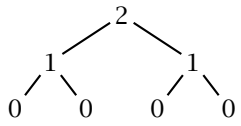
Aufgabe 7.34 Gibt es Listen über $\mathbb{N}$, die gemäß der Prä- oder Postlinearisierung keine Bäume darstellen?

Aufgabe 7.36 Verschränkte Rekursion kann stets auf einfache Rekursion zurückgeführt werden. Überzeugen Sie sich davon am Beispiel der Prozedur *eqset*, indem Sie eine semantisch äquivalente Prozedur deklarieren, die nur einfache Rekursion verwendet.

Aufgabe 7.37 Deklarieren Sie eine Prozedur *direct* : *tree* → *tree*, die zu einem Baum einen gerichteten Baum liefert, der die gleiche Menge darstellt. Verwenden Sie die polymorphe Sortierprozedur aus Aufgabe 5.14 auf S. 107 und die Vergleichsprozedur *compareTree* aus § 7.1.3.

Aufgabe 7.38 (a) Nehmen Sie an, dass finitäre Mengen durch gerichtete Bäume dargestellt werden. Deklarieren Sie Prozeduren des Typs *tree* → *tree* → *tree*, die zu zwei Mengen X, Y die Vereinigung $X \cup Y$, den Schnitt $X \cap Y$ und die Differenz $X - Y$ liefern.

Aufgabe 7.43 Schreiben Sie eine Prozedur *ltrd* : *int* → *int ltr*, die zu $n \geq 0$ einen balancierten Binärbaum der Tiefe n liefert, dessen Teilbäume mit ihrer Tiefe markiert sind. Für $n = 2$ soll *ltrd* den Baum



liefern. Verwenden Sie die Prozedur *iterup*.

Aufgabe 7.52 Schreiben Sie eine Prozedur *prep* : α *ltr* → α *list*, die die Präprojektion eines markierten Baums liefert.

Aufgabe 7.53 Schreiben Sie eine Prozedur *pop* : α *ltr* → α *list*, die die Postprojektion eines markierten Baums liefert.

Aufgabe 7.54 Die **Grenze** eines markierten Baums ist die Liste der Marken seiner Blätter, in der Ordnung ihres Auftretens von links nach rechts und mit Mehrfachauftreten. Die Grenze des Baums *t3* ist $[2, 7, 7, 4, 2, 7, 7]$. Schreiben Sie eine Prozedur *frontier* : α *ltr* → α *list*, die die Grenze eines Baums liefert.