

Vorname

Nachname

Matrikel-Nr

Schreiben Sie jede Lösung möglichst auf die Vorderseite eines neuen Blattes (und lassen Sie die Rückseiten Ihrer Lösungsblätter leer). Erreichbar sind 100 Punkte (50 für Note 4,0).

**Aufgabe 1** (30 Punkte): In einem Gentle-Programm sei folgender Typ definiert:

```
1 'type' INTLISTE
2   leer
3   k(Element: INT, Rest: INTLISTE)
```

Schreiben Sie ein Prädikat entsprechend der folgenden Spezifikation:

```
4 'condition' enthaeltDoppelgaenger(Liste: INTLISTE)
5   -- Gelingt, wenn die Liste mindestens einen "Doppelgaenger" enthaelt,
6   -- d.h. wenn mindestens ein INT-Wert mehr als einmal in der List
7   -- vorkommt.
```

Diese Aufgabe läßt sich besonders leicht lösen, wenn man ein geeignetes "Hilfsprädikat" programmiert und in der Vereinbarung des Prädikats `enthaeltDoppelgaenger` aufruft.

**Aufgabe 2** (20 Punkte): In einem Gentle-Programm sei folgender Typ definiert:

```
8 'type' BINBAUM
9   lb          -- leerer Baum
10  b(Element: INT, Links: BINBAUM, Rechts: BINBAUM)
```

Schreiben Sie ein Prädikat entsprechend der folgenden Spezifikation:

```
11 'condition' sindGleich(BINBAUM, BINBAUM)
12   -- Gelingt, wenn die beiden Parameter genau gleich sind.
```

**Aufgabe 3** (30 Punkte): Auch diese Aufgabe bezieht sich auf den Typ `BINBAUM`, der in der vorigen Aufgabe definiert wurde. Schreiben Sie ein Prädikat entsprechend der folgenden Spezifikation:

```
13 'condition' sindUngleich(BINBAUM, BINBAUM)
14   -- Gelingt, wenn die beiden Parameter nicht genau gleich sind.
```

**Aufgabe 4** (20 Punkte): Geben Sie eine kontextfreie Grammatik (d.h. eine Typ2-Grammatik) an für die "Sprache der int-Listen". Jedes Wort dieser Sprache hat folgende Form:

$(e_1, e_2, \dots, e_n)$

Jedes Wort stellt eine nicht-leere Liste von Elementen dar. Es muss mit einer öffnenden runden Klammer beginnen und mit einer schliessenden runden Klammer enden. Zwischen den Klammern dürfen ein oder mehrere, durch Kommas voneinander getrennte Elemente stehen. Jedes Element ist entweder eine vorzeichenlose Ganzzahl (z.B. 17 oder 0 oder 12345678) oder eine Liste. Es folgen ein paar Beispiele von Worten, die zur Sprache der int-Listen gehören:

```
1 (12, 345, 0)
2 (123, (1, 2, 4), 456)
3 ((1, 2), (3, 4, 5), (1, 2, 3, 4))
4 ( ((1, 2), (3, 4)), (5, 6))
5 (((((17))))))
6 (1)
7 ((3), (3), (3))
```

Es folgen ein paar Zeichenketten, die **nicht** zur Sprache der int-Listen gehören:

```
8  (1, 2, 3           // Schliessende Klammer fehlt
9  1, 2, 3)          // Oeffnende Klammer fehlt
10 1, 2, 3           // Beide Klammern fehlen
11 (1, 2, 3,)         // Ein Komma zu viel
12 (1,,,)            // Drei Kommas zu viel
13 (A, B, C)          // Buchstaben sind nicht erlaubt
14 {1, 2, 3}          // Nur runde Klammern sind erlaubt
15 ()                 // Leere Liste ist nicht erlaubt
16 (()), (), ()      // Leere Listen sind nicht erlaubt
17 ...
```

Ihre Grammatik soll auf folgender Menge von (dreizehn) Endsymbolen beruhen:

$T = \{ 0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ ( \ ) \ , \} \}$

Die Zwischensymbole können Sie frei wählen.

**Lösung 1 (30 Punkte):**

```
1 'type' INTLISTE
2   leer
3   k(Element: INT, Rest: INTLISTE)
4
5 'condition' enthaeltDoppelgaenger(Liste: INTLISTE)
6   -- Gelingt, wenn die Liste mindestens einen "Doppelgaenger" enthaelt,
7   -- d.h. wenn mindestens ein INT-Wert mehr als einmal in der List
8   -- vorkommt.
9   'rule' enthaeltDoppelgaenger(k(N, R)):
10     enthaelt(R, N).
11 'rule' enthaeltDoppelgaenger(k(N, R)):
12     enthaeltDoppelgaenger(R).
13
14 'condition' enthaelt(Liste: INTLISTE, Element: INT)
15 'rule' enthaelt(k(N1, R), N2):
16     eq(N1, N2).
17 'rule' enthaelt(k(N1, R), N2):
18     enthaelt(R, N2).
```

**Lösung 2 (20 Punkte):**

```
19 'type' BINBAUM
20   lb
21   b(Element: INT, Links: BINBAUM, Rechts: BINBAUM)
22
23 'condition' sindGleich(BINBAUM, BINBAUM)
24   -- Gelingt, wenn die beiden Parameter genau gleich sind.
25 'rule' sindGleich(lb, lb):.
26 'rule' sindGleich(b(N1, L1, R1), b(N2, L2, R2)):
27     eq(N1, N2)
28     sindGleich(L1, L2)
29     sindGleich(R1, R2)
```

**Lösung 3 (30 Punkte):**

```
30 'condition' sindUngleich(BINBAUM, BINBAUM)
31   -- Gelingt, wenn die beiden Parameter nicht genau gleich sind.
32 'rule' sindUngleich(lb, b(N, L, R))
33 'rule' sindUngleich(b(N, L, R), lb)
34 'rule' sindUngleich(b(N1, L1, R1), b(N2, L2, R2)):
35     ne(N1, N2).
36 'rule' sindUngleich(b(N1, L1, R1), b(N2, L2, R2)):
37     sindUngleich(L1, L2).
38 'rule' sindUngleich(b(N1, L1, R1), b(N2, L2, R2)):
39     sindUngleich(R1, R2).
```

**Lösung 4 (20 Punkte):**

```
40 Liste    -> '(' Elemente ')'      -- Liste ist das Startsymbol
41 Elemente -> Element
42 Elemente -> Element, Elemente
43 Element   -> Zahl
44 Element   -> Liste
45
46 Ziffer    -> 0
47 Ziffer    -> 1
48 Ziffer    -> 2
49 Ziffer    -> 3
50 Ziffer    -> 4
51 Ziffer    -> 5
52 Ziffer    -> 6
53 Ziffer    -> 7
54 Ziffer    -> 8
55 Ziffer    -> 9
56
57 Zahl      -> Ziffer
58 Zahl      -> Ziffer Zahl
```