Institut für Informatik
Prof. Dr. Peter Thiemann
Jochen Walter

Georges-Köhler-Allee 79
D-79110 Freiburg i. Br.

Freiburg, den 25. Januar 2001

Informatik 1, WiSe 2000/2001

Übungsblatt 11
Programmierprojekt 2

Das zweite Programmierprojekt läuft bis zum 10.2.2001, 12 Uhr.
In diesem Zeitraum werden in den Übungs- und Programmiergruppen Probleme im Zusammanhang mit dem Programmierprojekt besprochen.

Die Aufgaben auf diesem Blatt werden alleine bearbeitet. Bevor `drscheme` *gestartet werden kann, muß* `setup lang` *eingegeben werden. Der Sprachumfang muß auf „Full Scheme“ eingestellt werden.*

Die Abgabe erfolgt bis zum 9.2.2001 über das WWW. Informationen dazu finden sich ab dem 5.2.2001 auf den Web-Seiten zur Vorlesung. Die Lösungen aller Teilaufgaben müssen in einer Datei stehen. Jeder Teilnehmer kann mehrere Versionen seines Projektes abgeben. Die letzte Abgabe zählt.

Bei der Abgabe wird auf syntaktische Korrektheit der Programme geachtet. Programme mit syntaktischen Fehlen werden zurückgewiesen. Es empfiehlt sich deshalb, mit der Abgabe nicht bis zum Ende der Frist zu warten, sondern bereits Teillösungen abzugeben, um sich davon zu überzeugen, daß das Programm von dem Abgabeprogramm akzeptiert wird.

Das abgegebene Programm muß im ASCII-Format abgespeichert sein und sich unter der auf den Unix-Workstations im Rechnerpool installierten Version von DrScheme ausführen lassen. Falls Sie auf einem anderen Rechner (z.B. unter MS Windows) entwickeln, müssen Sie dafür Sorge tragen, daß Ihre Programmdatei diese Bedingungen erfüllt.

Das von Ihnen abgegebene Programm sollte für jede Funktion eine vollständige Spezifikation im Stil der in der Vorlesung angegeben Scheme-Funktionen haben. Punkte werden getrennt für Programmcode und Dokumentation vergeben.

Die Vorführung des Projektes in der Übungsgruppe und Erklärung der Funktionsweise ist Teil des Projekts.

Aufgabe 1 (0C+0D):

Besorgen Sie sich die Datei `program-2a.ss` von der Web-Seite der Vorlesung. Legen Sie eine Datei für Ihr Projekt an. Diese Datei soll einen Kommentar mit Ihrem Namen und Ihrer e-mail-Adresse enthalten. Laden Sie die Datei `program-2a.ss` mit der Zeile `(load "program-2a.ss")` in Ihre Programmdatei. (Ändern Sie also insbesondere nicht den vorgegebenen Code ab!)

Aufgabe 2 (1C+1D):

Unter einem Ring versteht man in den Mathematik einen ADT mit dem Rangalphabet (Σ, σ), wobei $\Sigma = \{\underline{+}, \underline{-}, \underline{\cdot}, \underline{0}\}$ und $\sigma(\underline{+}) = 2, \sigma(\underline{-}) = 2, \sigma(\underline{\cdot}) = 2, \sigma(\underline{0}) = 0$ und dessen Axiomenmenge Q wie folgt aussieht:

$$
\begin{aligned}
(a \underline{+} b) \underline{+} c &= a \underline{+} (b \underline{+} c) \\
\underline{0} \underline{+} a &= a \underline{+} \underline{0} = \underline{0} \\
a \underline{+} (\underline{0} \underline{-} a) &= (\underline{0} \underline{-} a) \underline{+} a = \underline{0} \\
b \underline{+} b &= b \underline{+} a \\
(a \underline{\cdot} b) \underline{\cdot} c &= a \underline{\cdot} (b \underline{\cdot} c) \\
a \underline{\cdot} (b \underline{+} c) &= a \underline{\cdot} b \underline{+} a \underline{\cdot} c \\
(a \underline{+} b) \underline{\cdot} c &= a \underline{\cdot} c \underline{+} b \underline{\cdot} c
\end{aligned}
$$

Eine wichtige Implementierung dieses ADT ist die Σ-Algebra $(\mathbb{Z}, \alpha)$ mit $\alpha(\underline{+}) = +, \alpha(\underline{-}) = -, \alpha(\underline{\cdot}) = \cdot, \alpha(\underline{0})() = 0$.

In dem vorgegebenen Scheme-Code gibt es eine Definition für die Struktur `ring` sowie eine einen Wert `numbers` dieses Typs, der der Implementierung $(\mathbb{Z}, \alpha)$ entspricht.

Andere wichtige Implementierungen sind die Restklassenringe $\mathbb{Z}_n$, d.h. die Σ-Algebren $\mathbb{Z}_n = (\{0, 1, \dots, n-1\}, \zeta_n)$ mit $\zeta_n(\underline{+})(a, b) = (a+b) \bmod n, \zeta_n(\underline{-})(a, b) = (a-b) \bmod n, \zeta_n(\underline{\cdot})(a, b) = (a \cdot b) \bmod n, \zeta_n(\underline{0})() = 0$.

Schreiben Sie eine Funktion `make-z-n: number -> ring`, die für eine gegebene Zahl $n \in \mathbb{N} \setminus \{0\}$ den entsprechenden Restklassenring zurückgibt. Der Name des Rings soll dabei `zet-`n lauten.

Aufgabe 3 (3C+3D):
Für einen Ring R nennt man Ausdrücke der Form $a_n \cdot x^n + a_{n-1} \cdot x^{n-1} + \dots + a_1 \cdot x + a_0$ mit $a_i \in R$ Polynome. Ist $a_n \neq 0$, dann nennt man n den Grad des Polynoms. (Falls $a_n = 0$ ist, dann ist der Grad des Polynoms gleich dem Grad von $a_{n-1} \cdot x^{n-1} + a_{n-2} \cdot x^{n-2} + \dots + a_1 \cdot x + a_0$.) Den Grad des Nullpolynoms, bei dem alle Koeffizienten $a_i = 0$ sind, definieren wir zu 0.

Im vorgegeben Scheme-Code gibt es eine Struktur `real-poly`. Schreiben Sie mit ihrer Hilfe eine Funktion `make-polynom: ring number (number -> ring) -> polynom`. Dabei erzeugt `(make-polynom r n f)` ein Polynom (d.h. eine `real-poly`-Struktur) mit Koeffizientenring `r` und oberer Abschätzung `n` für den Grad, dessen Koeffizienten a_i durch `(f i)` $(0 \leq i \leq n)$ gegeben sind. Verwenden Sie dazu Message-passing laut Vorlesung.

Aufgabe 4 (1C+1D):
Schreiben Sie drei weitere Funktion: `poly-degree: polynom -> number` liefert für ein Polynom seinen Grad zurück, `poly-ring: polynom -> ring` gibt für ein Polynom seinen Koeffizientenring zurück, und `poly-coeff: polynom number -> ring-element` liefert für ein Polynom und eine beliebige natürliche Zahl den Koeffizienten a_i zurück.

Aufgabe 5 (2C+2D):
Da Polynome Ringe sind, kann man auch Polynome betrachten, deren Koeffizienten wiederum Polynome sind. Dazu benötigt man eine entsprechende `ring`-Struktur, die mit der Funktion `make-polynom-ring` erzeugt werden kann.

In dieser Aufgabe soll eine Funktion `poly->string: polynom -> string` geschrieben werden, die aus einem Polynom eine Zeichenkette macht, die der üblichen mathematischen Schreibweise für ein Polynom entspricht. Dabei soll die Liste der Unbestimmten dem Polynom vorausgehen und von diesem durch einen Doppelpunkt abgetrennt werden. Beispiele sind:

```
Z:1Z^3+0Z^2-1Z^1+3Z^0
ZY:(1Y^1+1Y^0)Z^2-(1Y^0)Z^1+(2Y^0)Z^0
```

Falls ein Koeffizient aus einem Zahlenring stammt, soll er einfach so ausgegeben werden. Kommt ein Koeffizient hingegen aus einem Polynomring, soll dieser Koeffizient in Klammern geschrieben werden.

Diese Ausgabe kann noch (auf freiwilliger Basis) verbessert werden, indem man bei Koeffzienten aus einem Zahlenring die Ausgabe von `+1` und `-1` unterdrückt und stattdessen nur `+` und `-` schreibt. Weiter kann man den Exponenten einer Potenz weglassen, falls er 1 beträgt. Man schreibt also statt `Z^1` einfach `Z`. Falls er hingegen 0 beträgt, kann man die ganze Potenz weglassen, muß dann aber dafür sorgen, daß ein Koeffizient geschrieben wird. (Man schreibt also statt `Z^0 1`, falls die Koeffizienten aus einem Zahlenring kommen.)

Aufgabe 6 (3C+3D):
Geben Sie drei Scheme-Funktionen `poly-add`, `poly-sub` und `poly-mult` an, die alle die Signatur `polynom polynom -> polynom` haben. Diese drei Funktionen erwarten jeweils zwei Polynome über demselben Koeffizientenring und liefern die Summe, Differenz und Produkt dieser Polynome zurück.