ICOM 4036 – Programming Languages

Programming Assignment 2 – Symbolic Algebra in Scheme

Due on Tuesday December 7, 2003

Problem Statement

In this Problem Set you will develop several Scheme functions to manipulate symbolic polynomials of a single variable. A polynomial will be represented as a list consisting of a variable and a list of terms. Each term will be represented as a list including a coefficient and a degree. For instance the following polynomial:

$$5x^2 + 3x + 7$$

Will be represented by the following list:

```
('x   (5 2) (3 1) (7 0))
```

Notice that lists that represent polynomials may not have any terms with zero coefficients. Also, each list representing a polynomial should only have one term of a given degree. Moreover, the terms should be arranged in decreasing order of their degree.

A. Constructor and Accessor Methods

Define a constructor method **`make-poly`** of two parameters **var** and **terms** that returns a polynomial on variable **var** with a list of terms given by **terms**.

Define accesor methods **`get-var`** and **`get-terms`** to extract the variable and list of terms of a give polynomial.

B. Symbolic Arithmetic on Polynomials

Define a function **`+poly`** to that gets two polynomials represented as described above and returns the polynomial representing their sum.

Define a function **`-poly`** to that gets two polynomials represented as described above and returns the polynomial representing their subtraction.

Define a function **`*poly`** to that gets two polynomials represented as described above and returns the polynomial representing their product.

Define a function **`/poly`** to that gets two polynomials represented as described above and returns a list containing two polynomials representing their quotient and remainder. Use the long division method that you learned in high school.

C. Symbolic Calculus on Polynomials

Define a function **`ipoly`** to that gets a polynomial **p** represented as described above and a variable **v** and returns the polynomial representing the indefinite integral of **p** with respect to the argument variable **v**.

D. Evaluation and Pretty Printing of Polynomials

Define a function **`pp-poly`** that receives a polynomial and prints it to standard output in a form amenable for human reading.

Define a function **`eval-poly`** that receives a polynomial and a value and evaluates the polynomial at that value.

Design Guidelines and Software Development Platform

Make sure you organize and structure your code in a modular fashion using abstraction tools (e.g. functions) whenever your deem appropriate. Do not attempt to solve every function monolithically. Instead, try to capture common functionality into functions that can be applied multiple times. Make the most of Scheme's powerful first class procedure features.

You will carry out all your software development using the tools available at the Linux Academic Computing Lab (Amadeus). You may work on your personal Linux-based PC, but you must make sure that your code works with the version of MIT Scheme available at Amadeus as this is the system that the staff will use to grade your assignment.

Remember from the prontuario that your programming assignments will be graded according to the following late penalty policy:

Days Late	Percent Deduction
1 day late	25%
2 days late	50%
3 days late	100%

Programming assignments will be graded for both correctness and quality according to the following weights:

Criteria	Weight (%)
Correctness	60%
Design	20%
Efficiency	10%
Style & Documentation	10%

Remember that you can work in this assignment with at most one partner. However all members of a team are responsible for understanding all aspects of their implementation.