This assignment is worth 15% of the final mark of the declarative programming course. The code and documentation explaining your results are to be handed to the departmental secretary by Monday 17th May 2004. Assignments handed in late will be marked down by 3 marks (out of 15) per day. No assignments will be accepted after Friday 21st May 2004.

The Department of Computer Science and AI takes a very serious view on plagiarism. Refer to the departmental website on plagiarism for more details:

http://www.cs.um.edu.mt/resources/plagiarism/

You are to solve one of the following two problems.

1. Computers were created to carry out number crunching, and such operations are simple to program using traditional computer languages. A more challenging task is that of performing operations on symbolic data, such as a mathematical algebraic expression. In this question you will be aiming at building a simple symbolic differentiator, which takes an arithmetic expression and a variable as input, and produces another expression which is the result of differentiating the given expression by the given variable. For example, differentiating $3x^2$ with respect to $x$, should return $6x$.

   (a) Define a Haskell datatype to represent arithmetic expressions which may use addition, subtraction, multiplication, division, unary minus, exponentiation, and may have constants and variables.

   (b) Make your datatype an instance of the typeclass `Show`, to be able to pretty-print your arithmetic expressions.

   (c) You may recall some basic rules of differentiation:

   $\frac{dc}{dx} = 0$ where $c$ is a numeric constant

   $\frac{dy}{dx} = 0$ where $y \neq x$

   $\frac{dx}{dx} = 1$

   $\frac{d(f + g)}{dx} = \frac{df}{dx} + \frac{dg}{dx}$

   $\frac{d(f - g)}{dx} = \frac{df}{dx} - \frac{dg}{dx}$

   $\frac{d(-f)}{dx} = -\frac{df}{dx}$

   $\frac{d(f^n)}{dx} = n \times f^{(n-1)} \times \frac{df}{dx}$ where $n$ is a numeric constant

   $\frac{d(f \times g)}{dx} = \frac{df}{dx} \times g + f \times \frac{dg}{dx}$

   $\frac{d(f \div g)}{dx} = \frac{df}{dx} \div g + f \div \frac{dg}{dx}$

   Write a function which given an expression $e$ and variable $x$, returns an expression representing $\frac{de}{dx}$.

   (d) Various laws of arithmetic can be used to simplify expressions. For example, $x \times 0 = 0$. Similarly, $1 \div 2$ could be replaced by 0.5. Write a function which implements a number of such rules to simplify expressions. There are many rules one could use in the implementation. It is up to you to identify a number of rules, and implement them correctly.

2. You have already encountered Turing Machines in previous courses. The task in this question is to implement a simple Turing Machine simulator in Haskell.
(a) Define datatypes to represent the alphabet, the states, the tape of a Turing Machine. Use these to define datatype to represent Turing Machines. Recall that to describe a Turing Machine, you need to identify its (i) alphabet, (ii) states, (iii) initial state, (iv) final states, (v) the transition relation. Read about Haskell records to make the description of Turing Machines better structured.

(b) Make your Turing Machine datatype an instance of the type class `Show`, to be able to display a textual description of a Turing Machine.

(c) Write a function which given a Turing Machine and its tape, simulates one step of the Turing Machine.

(d) Write a function, which given a Turing Machine and an input, simulates the machine until termination, returning the resulting tape.