Symbolic algebra was one of the original application areas for Lisp, the ancestor of Scheme. In this assignment, you will write a procedure that solves simple symbolic equations. Your procedure should be defined in the file solver.scm.

The procedure solver has two arguments: a symbol `var` and an equality expression `eqn`. The variable should appear once somewhere in the equality expression, either as part of the left operand or the right operand. The result of `(solver var eqn)` should be another equality where `var` appears alone as the left operand and the right operand is the original equation, solved for `var`.

The top level of the input equation `eqn` is a list of the form `(= exp1 exp2)`. The two expressions `exp1` and `exp2` are legal Scheme expressions formed from atoms, numbers, and procedure calls involving +, -, *, and / (only). You should assume that each of these functions is binary (ie, has exactly two arguments). You don't need to worry about division by zero.

If there is exactly one occurrence of `var` in `eqn`, then solve the equation and return the solution. Otherwise, return the string "Too Hard". Except for checking for the right number of occurrences of `var`, you can assume that the input is well-formed.

The key to designing this procedure is to unwind the equation, one operator at a time, building a new and equivalent equation at each step.

The file expression.scm is provided to you, and contains definitions for a constructor and accessor functions for binary expressions.

1. Write the solver procedure. The file hw4-test-it.scm contains several test cases with expected solutions.

2. Once your solver is running without major problems, you can run it again with run-solver-from-file.scm. This file contains a procedure that reads equations from a file, passes them to your solver, and then writes the results to a text file and a plot file. The file automatically runs with the equations in x.txt, producing the text file x-solved.txt and the dot input file x-solved.dot. You can convert the dot file to postscript by running dot on it (batch command file rundot-x.bat). Review the resulting plot to see what your expressions actually look like as trees.