Computer Science & Engineering 341
Assignment 7: Miranda Project May 12, 1998
Due: May 22, 1998

Turn in a listing of your script both questions, and sample output showing your function working correctly.

1. Write and test a Miranda function `multiply` that multiplies two polynomials in x and returns the result. The two polynomials should be represented as strings. For example:

   ```
   multiply "x^3 + x^2 + x + 1"  "x - 1"    =>
   "x^4 - 1"
   ```

   Here are some polynomial pairs to test your function on:

   ```
   "x^3 + x^2 + x + 1"
   "x - 1"

   "-3*x^3 + x + 5"
   "0"

   "x - 1 + x^3"
   "-5"

   "-10*x^2 + 100*x + 5"
   "x^9999 - x^7 - x^5 + x + 3"
   ```

   In general each string representing a polynomial will consist of a series of terms, separated by + or - signs. Each term will have an integer coefficient (assumed to be 1 if absent), followed by the letter x, followed by the symbol ^, followed by an integer exponent. If the ^ is missing, the exponent defaults to 1. If the x and exponent are both missing, the term is a constant.

   You don't have to check for bad inputs.

   Hint: use a different internal representation of a polynomial for doing the multiplication, for example, a list of (coefficient exponent) pairs. If you do this, the top-level function is
   `multiply p q = tostring (polymult (tolist p) (tolist q))`
   where `tolist` and `tostring` convert to and from the internal representation.

2. Make up, write, and test your own Miranda script that uses infinite data structures in an interesting way. If you can't think of anything interesting, make up a program that uses them in an uninteresting way . . . you don't need to do anything big to get full credit for this question, but I'm hoping a few students will do something fun with it. (Some samples from another course are in the directory `~borning/505/miranda/infinite` on orcas.) One rather challenging possibility would be to allow infinite polynomials in your polynomial multiplier. (If you do this, require that the polynomials be entered with the terms sorted by exponent, smallest first, unlike in question 1.)