CS 403, Assignment 1

Due on Sept. 26 at 11:59 pm

Any polynomial $a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n$ can be represented in HASKELL by the list `[ a0, a1, a2, ..., an ]` of its coeficients. You may take `[]` (the empty list) to be a representation of the identity element for the `+` operation.

Suppose we want to compute with polynomials; define the following functions using the suggested list representation.

1. A function `polyEval x p` that evaluates the polynomial (represented by list) `p` at `x`; for example,

   ```
   polyEval 2.0 [1.0, 2.0, 3.0]
   ```

   should evaluate to 17.0. Use Horner's evaluation scheme:

   $$a_0 + x \cdot (a1 + \cdots + x \cdot (a_{n-1} + x \cdot a_n) \cdots)$$

   and a list fold (with a custom folding operation) in your implementation.

2. A function `polyAdd p q` that returns a representation of the sum of the polynomials represented by `p` and `q`; for example,

   ```
   polyAdd [1.0, 2.0, 3.0] [2.0, 3.0]
   ```

   should evaluate to `[3.0, 5.0, 3.0]`. Try to use `zip` and a map in your implementation and compare such a version with the (straightforward) inductive definition.

3. A function `polyMult p q` that returns a representation of the product of the polynomials represented by p and q; for example,

   ```
   polyMult [1.0, 2.0, 3.0] [2.0, 3.0]
   ```

   should evaluate to `[2.0, 7.0, 12.0, 9.0]`. Use the fact that, when $n > 0$, $a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n$ is expressible as $a_0 + x \cdot (a_1 \cdot + a_2 \cdot x + \cdots + a_n \cdot x^{n-1})$ and thus provide a recursive definition that uses `polyAdd` and a map.

You can consider that we are only interested in polynomials with `Float` coefficients. However attempting to provide general, explicit type definitions for your functions is encouraged and success on the matter will result in bonus marks.

Please observe the implementation requirements (but do note the obvious difference between "use" and "try to use").

Submission guidelines

Provide your scripts and the plain English answers to questions as appropriate. Also provide (as comments) sessions listings that test your functions.

Submit a single plain text file that can be loaded in the HASKELL interpreter. It would be nice if you can submit a literate script (`.lhs` extension), but this is not required. Provide your submission by email.

Recall that assignments can be solved in groups (of maximum three students), and a single solution per group should be submitted. Also recall that a penalty of 10% per day will be applied to late submissions until the solution is released.