Homework 5: Streams Due: Wed Oct. 26, 2022

Introduction

In this assignment, you will work with streams to evaluate power series.
Consider the series $s(x) = a_0 + a_1 x + a_2 x^2 + ...$ We can represent this series by its finite or infinite sequence of coefficients $(a_0, a_1, a_2, ...)$. We will view this sequence as a stream.

Specification

For all functions below, use memoized streams. A series is represented by a nonempty stream, which may be finite or infinite. Compute only as much of the result stream as needed.

1. Write a function `addSeries` that takes two streams of coefficients for the series *s(x)* and *t(x)* and returns the stream of coefficients for the sum *s(x) + t(x)*.
For example, given $1+2x+3x^2+...$ and $2+6x+9x^2+...$, the result will be $3+8x+12x^2+...$

2. Write a function `prodSeries` that takes two streams of coefficients for the series *s(x)* and *t(x)* and returns the stream of coefficients for the product $s(x) \cdot t(x)$. (Review polynomials)
For example, given $1+2x+3x^2+...$ and $2+6x+9x^2+...$, the result will be $2+10x+27x^2+...$

Hint: Write one of the series as $s(x) = a_0 + x\, s_1(x)$, where $s_1(x)$ is another series. Then use distributivity to multiply $s(x) \cdot t(x)$ and map all operations to streams (how can you represent multiplying with *x*?). Delay the recursive computation of the result's tail until needed.

3. Write a function `derivSeries` that takes a stream of coefficients for the series *s(x)*, and returns a stream of coefficients for the derivative *s′(x)*.
For example, given $1+2x+3x^2+...$, the result will be $2+6x+...$,

4. Write a function `coeff` that takes a stream of coefficients for the series *s(x)* and a natural number n, and returns the array of coefficients of *s(x),* up to and including that of order n.
If the stream has fewer coefficients, return as many as there are.

5. Write a function `evalSeries` that takes a stream of coefficients for the series *s(x),* and a natural number n, and returns a closure. When called with a real number x, this closure will return the sum of all terms of *s(x)* up to and including the term of order n.

6. Write a function `applySeries` that takes a function f and a value v and returns the stream representing the infinite series *s(x)* where a_0 = v, and $a_{k+1} = f(a_k)$, for all $k \geq 0$.

7. Write a function `expSeries` with no arguments that returns the Taylor series for e^x, i.e., the coefficients are $a_k = 1/k!$ You may use `applySeries` with an appropriate closure.

8. Write a function `recurSeries`, taking two arrays, `coef` and `init`, assumed of equal positive length k, with coef = $[c_0, c_1, \ldots, c_{k-1}]$. The function should return the infinite stream of values a_i given by a k-order recurrence: the first k values are given by init: $[a_0, a_1, \ldots, a_{k-1}]$; the following values are given by the relation $a_{n+k} = c_0\, a_n + c_1\, a_{n+1} + \ldots + c_{k-1}\, a_{n+k-1}$ for all $n \geq 0$.