Exercise 3.54.

Define a procedure `mul-streams`, analogous to `add-streams`, that produces the elementwise product of its two input streams. Use this together with the stream of `integers` to complete the following definition of the stream whose nth element (counting from 0) is $n+1$ factorial:

```
(define factorials (cons-stream 1 (mul-stream <??> <??>)))
```

Answer.

`Mul-streams` is defined in a way much similar to `add-streams`:

```
(define (mul-streams s1 s2)
  (stream-map * s1 s2))
```

Besides, we know that factorials is computed based on the observation that:

- $n!$ is equal to n times $(n-1)!$ for any positive integer n, and
- $0! = 1$

This indicates that `factorials` is a stream begining with 1, such that the rest of the stream can be generated by multiplying `integers` to the `factorials` itself, as figure 1 shows. Hence, we can define the stream of `factorials` to be

```
(define factorials (cons-stream 1 (mul-stream integers factorials)))
```

	1	2	3	4	5	6	7	...	= `integers`
	0!	1!	2!	3!	4!	5!	6!	...	= `factorials`
1	1	2	6	24	120	720	5040	...	= `factorials`

Figure 1. Process of generating elements of the stream `factorials`.

Email address: informlarry@gmail.com