4.70 What is the purpose of the let bindings in the procedures add-assertion! and add-rule! ? What would be wrong with the following implementation of add-assertion!?

Q:

```
(define (add-assertion! assertion)
   (store-assertion-in-index assertion)
   (set! THE-ASSERTIONS
         (cons-stream assertion THE-ASSERTIONS))
   'ok)
```

Real form:

```
(def (add-ass! ass)
     (store-ass-in-index ass)
     (let ((old-ass THE-ASS))
       (set! THE-ASS
             (cons-stream ass old-ass))
       'ok))
```

Hint: Recall def of infinite stream of ones from sect. 3.5.2: (def ones (cons-stream 1 ones))

A: Streams are delayed, and like the ones, without the let bindings THE-ASSERTIONS would be set to a infinite stream of the one assertion. With the let bindings, the second element of the stream (which is delayed) will later be forced to old-assertions, as desired.