Streams as a data structure - 2/09

(cons-stream a b) ⇒ stream, say 's'

(stream-car s) ⇒ a.
(stream-cdr s) ⇒ b.
} → selectors

constructor

the-empty-stream ⇒ '().

(stream-null? s) ⇒ (eq? s the-empty-stream)

```
(define (stream-enum low high)
    (if (> low high)
        the-empty-stream
        (cons-stream low (stream-enum
                          (+ low 1) high))))

(define (stream-filter pred s)
    (cond ((stream-null? s) the-empty-stream)
          ((pred (stream-car s))
           (cons-stream (stream-car s)
                        (stream-filter (stream-cdr s))))
          (else (stream-filter (stream-cdr s)))))

(stream-car (stream-cdr (stream-filter prime?
                         (stream-enum 10K 1M)))).
```

This is supposed to be more efficient than what we wrote earlier.

So far it looks the same as list

(cons-stream a b) ⇒ (cons a (delay b))
(stream-car s) ⇒ (car s)
(stream-cdr s) ⇒ (force (cdr s))
forces the delay.

when you pass something to a 'delay', the result is a 'promise'
when you pass a 'delay' to 'force', it forces the promise (similar to the notion of a promise on a note). call the promise.

If we want to delay the evaluation of an expression, put it inside a lambda so that it is evaluated when its invoked

(delay <expr>) ⇒ (lambda () <expr>).
(force promise) ⇒ (promise).

(stream-car (stream-cdr (stream-filter
prime? (stream-ei 10k 1M))))

(cons 10k (delay (stream-ei 10k+1 1M)))
(after skipping one step)

this is now passed to stream-filter.
is 10k a prime - applying predicate
predicate is false - goes to else branch
- applies predicate to car.

its not being evaluated (whatever is in delay) because. it wasn't called so far.

(stream-ei 10k+1 1M).

⇒ (cons 10k+1 (delay (stream-ei 10k+2 1M)))

pass this to stream-filter again. ~~10007~~

10k+7 - first prime

so similar things till there

⇒(cons 10007 (delay (stream-ei 10k+8 1M))).

stream-car s.

(cons 10008 (delay (stream-filter (cons 10009 14)

supplied to stream-cdr in

(stream-car (stream-cdr (stream-filter prime? (stream-ei 10k 1M))))

when stream-cdr forces it, calls stream-filter on the stream - checks the first number.

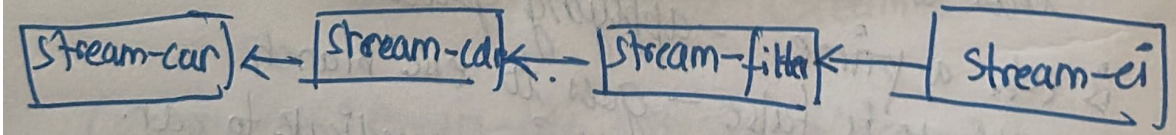
(checks 10k+8 - not prime)

(checks 10k+9 - prime).

stream-cdr gives out

(cons 10k+9 (delay (stream-filter (cons 10k+10 1M))))

stream-car of this : 10009



The order in which things are happening is diff from how the computation happens