Let's add type inference to our compiler

Example 1

(defn (incr x) (+ x 1))

(incr input)

Example 2

(defn (id x) x)

(let* ((a1 (id 7))
       (a2 (id true)))
  true)
Example 3

(defn (f it x)
  (+ (it x) 1))

(defn (incr z)
  (+ z 1))

(f incr 10)
Example 4

;; --- an API for lists ---------------------------------------------
(defn (nil) (as (forall (a) (-> () (list a)))
  false)

(defn (cons h t) (as (forall (a) (-> (a (list a)) (list a)))
  (vec h t))

(defn (head l) (as (forall (a) (-> ((list a)) a)))
  (vec-get l 0))

(defn (tail l) (as (forall (a) (-> ((list a)) (list a)))
  (vec-get l 1))

(defn (isnil l) (as (forall (a) (-> ((list a)) bool)))
  (= l false))

;; --- computing with lists ---------------------------------------
(defn (length xs)
  (if (isnil xs)
    0
    (+ 1 (length (tail xs)))))

(defn (sum xs)
  (if (isnil xs)
    0
    (+ (head xs) (sum (tail xs)))))

(let (xs (cons 10 (cons 20 (cons 30 (nil))))))
(vec (length xs) (sum xs)))