Facultad de Ciencias, UNAM
Lenguajes de Programación
Tarea 7

Rodríguez Campos Erick Eduardo
Rubí Rojas Tania Michelle

31 de enero de 2021

1. Da la derivación de las siguientes expresiones usando las reglas de semántica operacional para FAE, vistas en clase:

   a) `{- {{fun {x} x} 2 } {+ 3 5}}`
   Solución:

$$\dfrac{\dfrac{\texttt{\{fun \{x\} x\},}\ \varnothing \Rightarrow \langle x, x, \varnothing\rangle \qquad \texttt{2,}\ \varnothing \Rightarrow \hat{2} \qquad \texttt{x, [x} \leftarrow \texttt{2]} \Rightarrow \hat{2}}{\texttt{\{\{fun \{x\} x\} 2\},}\ \varnothing \Rightarrow \hat{2}} \qquad \dfrac{\texttt{3,}\ \varnothing \Rightarrow \hat{3} \qquad \texttt{5,}\ \varnothing \Rightarrow \hat{5}}{\texttt{\{+ 3 5\},}\ \varnothing \Rightarrow \hat{8}}}{\texttt{\{- \{\{fun \{x\} x\} 2\} \{+ 3 5\}\},}\ \varnothing \Rightarrow \texttt{-6}}$$

   b) `{{{fun {x} {fun {y}{+ x y}}} 2} 3}`
   Solución:

$$\dfrac{\dfrac{\texttt{A} \qquad \texttt{B} \qquad \texttt{C}}{\texttt{D}} \qquad \texttt{H} \qquad \dfrac{\texttt{E} \qquad \texttt{F}}{\texttt{G}}}{\texttt{I}}$$

   donde

   - A = `{fun {x} {fun {y} {+ x y}}}`, $\varnothing \Rightarrow \langle$`x, {fun {y} {+ x y}},` $\varnothing\rangle$
   - B = `2,` $\varnothing \Rightarrow \hat{2}$
   - C = `{fun {y} {+ x y}}[x` $\leftarrow \hat{2}$`]` $\Rightarrow \langle$ `y, {+ x y}, [x` $\leftarrow \hat{2}$`]`$\rangle$
   - D = `{{fun {x} {fun {y} {+ x y}}} 2},` $\varnothing \Rightarrow \langle$`y, {+ x y}, [x` $\leftarrow \hat{2}$`]`$\rangle$
   - E = `x, [x` $\leftarrow \hat{2}$`, y` $\leftarrow \hat{3}$`]` $\Rightarrow$ `[x` $\leftarrow \hat{2}$`, y` $\leftarrow \hat{3}$`](x)` $= \hat{2}$
   - F = `y, [x` $\leftarrow \hat{2}$`, y` $\leftarrow \hat{3}$`]` $\Rightarrow$ `[x` $\leftarrow \hat{2}$`, y` $\leftarrow \hat{3}$`](y)` $= \hat{3}$
   - G = `{+ x y}, [x` $\leftarrow \hat{2}$`, y` $\leftarrow \hat{3}$`]` $\Rightarrow \hat{5}$
   - H = `3,` $\varnothing \Rightarrow \hat{3}$
   - I = `{{{fun {x} {fun {y} {+ x y}}} 2} 3},` $\varnothing \Rightarrow \hat{5}$

2. Realiza el juicio de tipo para cada una de las siguientes expresiones, usa las reglas visitas en clase.

   (*a*)
```
{with {a : number 2}
      {+ a 2}}
```

$$\dfrac{\varnothing \vdash 2 : number \qquad \dfrac{[a \leftarrow number] \vdash a : number \qquad [a \leftarrow number] \vdash 2 : number}{[a \leftarrow number] \vdash \{+\ a\ 2\} : number}}{\varnothing \vdash \{with\{a : number\ 2\}\ \{+\ a\ 2\}\} : number}$$

   (*b*) `{fun {x : number} : number {+ x 2}}`

$$\dfrac{\dfrac{[x \leftarrow number] \vdash x : number \qquad [x \leftarrow number] \vdash 2 : number}{[x \leftarrow number] \vdash \{+\ x\ 2\} : number}}{\varnothing \vdash \{fun\{x : number\} : number\{+\ x\ 2\}\} : \{number \rightarrow number\}}$$

(c) `{{fun {x} {+ x 2}} {+ 3 4}}`

$$\dfrac{\dfrac{\dfrac{[x \leftarrow number] \vdash x : number \qquad [x \leftarrow number] \vdash 2 : number}{[x \leftarrow number] \vdash \{+\ x\ 2\} : number}}{\varnothing \vdash \{fun\{x : number\} : number\{+\ x\ 2\}\} : \{number \rightarrow number\}} \qquad \dfrac{\varnothing \vdash 3 : number \qquad \varnothing \vdash 2 : number}{\varnothing \vdash \{+\ 3\ 4\} : number}}{\varnothing \vdash \{\{fun\{x : number\} : number\{+\ x\ 2\}\}\ \{+\ 3\ 4\}\} : number}$$

(d)
```
{with {f : {number -> number} {fun {x : number} : number {+ x 2}}}
    {f {+ 3 4}}}
```

$$\dfrac{\dfrac{\dfrac{A \qquad B}{C}}{D} \qquad \dfrac{E \qquad \dfrac{F \qquad G}{H}}{I}}{J}$$

Donde:

A es $[x \leftarrow number] \vdash x : number$

B es $[x \leftarrow number] \vdash 2 : number$

C es $[x \leftarrow number] \vdash \{+\ x\ 2\} : number$

D es $\varnothing \vdash \{fun\{x : number\} : number\{+\ x\ 2\}\} : \{number \rightarrow number\}$

E es $[f \leftarrow \{number \rightarrow number\}] \vdash f : \{number \rightarrow number\}$

F es $[f \leftarrow \{number \rightarrow number\}] \vdash 3 : number$

G es $[f \leftarrow \{number \rightarrow number\}] \vdash 2 : number$

H es $[f \leftarrow \{number \rightarrow number\}] \vdash \{+\ 3\ 4\} : number$

I es $[f \leftarrow \{number \rightarrow number\}] \vdash \{f\{+\ 3\ 4\}\} : number$

J es $\varnothing \vdash \{with\{f : \{number \rightarrow number\}\{fun\ \{x : number\} : number\ \{+\ x\ 2\}\}\}\ \{f\{+\ 3\ 4\}\}\} : number$

(e)
```
{with {g {fun {x} {x 4}}}
    {g {fun {y} {- y 2}}}}
```

$$\dfrac{\dfrac{\dfrac{A \qquad B}{C}}{D} \qquad \dfrac{E \qquad \dfrac{\dfrac{F \qquad G}{H}}{I}}{J}}{K}$$

Donde:

A es $[x \leftarrow number] \vdash x : \{number \rightarrow number\}$

B es $[x \leftarrow number] \vdash 4 : number$

C es $[x \leftarrow number] \vdash \{x\ 4\} : number$

D es $\varnothing \vdash \{fun\{x : number\} : number\{x\ 4\}\} : \{number \rightarrow number\}$

E es $[f \leftarrow \{number \rightarrow number\}] \vdash g : \{number \rightarrow number\}$

F es $[f \leftarrow \{number \rightarrow number\}, y \leftarrow number] \vdash y : number$

G es $[f \leftarrow \{number \rightarrow number\}, y \leftarrow number] \vdash 2 : number$

H es $[f \leftarrow \{number \rightarrow number\}, y \leftarrow number] \vdash \{-\ y\ 2\} : number$

I es $[f \leftarrow \{number \to number\}] \vdash \{fun\{y\}\{-\ y\ 2\}\} : \{number \to number\}$

J es $[f \leftarrow \{number \to number\}] \vdash \{g\ \{fun\{y\}\{-\ y\ 2\}\}\} : number$

K es $\varnothing \vdash \{with\{g : \{number \to number\}\ \{fun\{x : number\} : number\{x\ 4\}\}\}\{g\ \{fun\{y\}\{-\ y\ 2\}\}\}\} : number$

(*f*)
```
{rec {f : {number -> number}
     {fun {x : number} : number
         {if0 x 1 {* n {f {- n 1}}}}}}
     {f 5}}
```

$$\dfrac{\dfrac{A \quad B}{C} \qquad \dfrac{\dfrac{D \quad E \quad \dfrac{F \quad \dfrac{G \quad \dfrac{H \quad I}{J}}{K}}{L}}{M}}{N}}{O}$$

Donde:

A es $[f \leftarrow \{number \to number\}] \vdash f : \{number \to number\}$

B es $[f \leftarrow \{number \to number\}] \vdash 5 : number$

C es $[f \leftarrow \{number \to number\}] \vdash \{f\ 5\} : number$

D es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash x : number$

E es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash 1 : number$

F es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash x : number$

G es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash f : \{number \to number\}$

H es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash n : number$

I es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash \{-\ n\ 1\} : number$

J es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash 1 : number$

K es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash \{f\{-\ n\ 1\}\} : number$

L es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash \{*\ n\{f\{-\ n\ 1\}\}\} : number$

M es $[f \leftarrow \{number \to number\},\ x \leftarrow number] \vdash \{if0\ x\ 1\{*\ n\ \{f\{-\ n\ 1\}\}\}\} : number$

N es $[f \leftarrow \{number \to number\}] \vdash \{fun\{x : number\} : number\{if0\ x\ 1\{*\ n\ \{f\{-\ n\ 1\}\}\}\}\} : \{numero \to numero\}$

O es $\varnothing \vdash \{\{rec\{f : \{number \to number\}\ \{fun\{x : number\} : number\ \{if0\ x\ 1\{*\ n\ \{f\ \{-\ n\ 1\}\}\}\}\}\}\}\ \{f\ 5\}\} : number$

3. Para cada una de las siguientes expresiones, realiza su inferencia de tipos generando las restricciones de tipo correspondientes

   a)
```
(define (potencia a b)
   (if (zero? b)
       1
       (* a (potencia a (sub1 b)))))
```

Solución: Primero, identificamos cada una de nuestras sub-expresiones y las enumeramos.

- $\boxed{0}$ `(define (potencia a b) (if (zero? b) 1 (* a (potencia a (sub1 b)))))`
- $\boxed{1}$ `(if (zero? b) 1 (* a (potencia a (sub1 b))))`
- $\boxed{2}$ `(zero? b)`

- $\boxed{3}$ `1`
- $\boxed{4}$ `(* a (potencia a (sub1 b)))`
- $\boxed{5}$ `(potencia a (sub1 b))`
- $\boxed{6}$ `(sub1 b)`

Luego, vamos a analizar el tipo de expresiones que encontramos.

- Para la cajita cero,
$$[[\ \boxed{0}\ ]] = [[a]] \times [[b]] \to [[\ \boxed{1}\ ]]$$
- Para la primer cajita,
$$\begin{aligned} [[\ \boxed{1}\ ]] &= [[\ \texttt{(if (zero? b) 1 (* a (potencia a (sub1 b))))}\ ]] \\ &= [[\ \texttt{(if } \boxed{\texttt{2}}\ \boxed{\texttt{3}}\ \boxed{\texttt{4}}\texttt{)}\ ]] \end{aligned}$$
  de donde
  - $[[\ \boxed{1}\ ]] = [[\ \boxed{3}\ ]]$
  - $[[\ \boxed{1}\ ]] = [[\ \boxed{4}\ ]]$
  - $[[\ \boxed{3}\ ]] = [[\ \boxed{4}]]$
  - $[[\ \boxed{2}\ ]] = \texttt{boolean}$
- Para la segunda cajita,
$$[[\ \boxed{2}\ ]] = [[\texttt{(zero? b)}]]$$
  de donde
  - $[[\texttt{(zero? b)}]] = \texttt{boolean}$
  - $[[\texttt{b}]] = \texttt{number}$
- Para la tercer cajita,
$$[[\ \boxed{3}\ ]] = [[1]] = \texttt{number}$$
- Para la cuarta cajita,
$$\begin{aligned} [[\ \boxed{4}\ ]] &= [[\texttt{(* a (potencia a (sub1 b)))}]] \\ &= [[\texttt{(* a } \boxed{5}]] \end{aligned}$$
  de donde
  - $[[\ \boxed{4}\ ]] = \texttt{number}$
  - $[[\texttt{a}]] = \texttt{number}$
  - $[[\ \boxed{5}\ ]] = \texttt{number}$
- Para la quinta cajita,
$$\begin{aligned} [[\ \boxed{5}\ ]] &= [[\texttt{(potencia a (sub1 b))}]] \\ &= [[\texttt{(potencia a } \boxed{6}\texttt{)}]] \end{aligned}$$
  de donde $[[\texttt{a} \to \boxed{6}]]$
- Para la sexta cajita,
$$[[\ \boxed{6}\ ]] = [[\texttt{(sub1 b)}]]$$
  de donde
  - $[[\texttt{(sub1 b)}]] = \texttt{number}$
  - $[[\texttt{b}]] = \texttt{number}$

Por lo tanto, el tipo de la función `potencia` es

$$\texttt{potencia: number number -> number}$$

donde a y b son ambos `number`.

b)
```
(define (suma l)
    (if (nempty? l)
        0
        (ncons (nfirst l) (suma (nrest l)))))
```

SOLUCIÓN: Primero, identificamos cada una de nuestras sub-expresiones y las enumeramos.

- $\boxed{0}$ `(define (suma l) (if (nempty? l) 0 (ncons (nfirst l) (suma (nrest l)))))`
- $\boxed{1}$ `(if (nempty? l) 0 (ncons (nfirst l) (suma (nrest l))))`
- $\boxed{2}$ `(nempty? l)`
- $\boxed{3}$ `0`
- $\boxed{4}$ `(ncons (nfirst l) (suma (nrest l)))`
- $\boxed{5}$ `(nfirst l)`
- $\boxed{6}$ `(suma (nrest l))`
- $\boxed{7}$ `(nrest l)`

Luego, vamos a analizar el tipo de expresiones que encontramos.

- Para la cajita cero,
$$[[\,\boxed{0}\,]] = [[l]] \to [[\,\boxed{1}\,]]$$
- Para la primer cajita,
$$\begin{aligned}[[\,\boxed{1}\,]] &= [[\texttt{(if (nempty? l) 0 (ncons (nfirst l) (suma (nrest l))))}]]\\ &= [[\texttt{(if }\boxed{2}\ \boxed{3}\ \boxed{4}\texttt{)}]]\end{aligned}$$
  de donde
  - $[[\,\boxed{1}\,]] = [[\,\boxed{3}\,]]$
  - $[[\,\boxed{1}\,]] = [[\,\boxed{4}\,]]$
  - $[[\,\boxed{3}\,]] = [[\,\boxed{4}\,]]$
  - $[[\,\boxed{2}\,]] = \texttt{boolean}$
- Para la segunda cajita,
$$[[\,\boxed{2}\,]] = [[\ \texttt{(nempty? l)}\ ]]$$
  de donde
  - $[[\texttt{(nempty? l)}]] = \texttt{boolean}$
  - $[[\text{b}]] = \texttt{nlist}$
- Para la tercer cajita,
$$[[\,\boxed{3}\,]] = [[0]] = \texttt{number}$$
- Para la cuarta cajita,
$$\begin{aligned}[[\,\boxed{4}\,]] &= [[\ \texttt{(ncons (nfirst l) (suma (nrest l)))}\ ]]\\ &= [[\texttt{(ncons }\boxed{5}\ \boxed{6}\texttt{)}]]\end{aligned}$$
  de donde
  - $[[\,\boxed{4}\,]] = \texttt{nlist}$
  - $[[\,\boxed{5}\,]] = [[\texttt{(nfirst l)}]] = \texttt{number}$
  - $[[\,\boxed{6}\,]] = [[\texttt{(suma (nrest l))}]]$
    - $[[\text{suma}]] = [[\texttt{(nrest l)}]] \to [[\,\boxed{6}\,]] = [[\,\boxed{7}\,]] \to [[\,\boxed{6}\,]] = \texttt{nlist}$

◇ $[[$`(nrest l)`$]] =$ `nlist`

◇ $[[$`l`$]] =$ `nlist`

Sin embargo, por el análisis de la primer cajita tenemos que $[[\boxed{3}]] = [[\boxed{4}]]$, pero

$$[[\boxed{3}]] = \texttt{number} \neq \texttt{nlist} = [[\boxed{4}]]$$

Por lo tanto, obtenemos una contradicción.

c)
```
(define (nfilter p l)
    (cond
        [(nempty? l) nempty]
        [(p (nfirst l)) (ncons (nfirst l) (nfilter p (nrest l)))]
        [else (nfilter p (nrest l))]))
```

Solución: Primero, identificamos cada una de nuestras sub-expresiones y las enumeramos.

- $\boxed{0}$ `(define (nfilter p l) (cond [(nempty? l) nempty] [(p (nfirst l)) (ncons (nfirst l) (nfilter p (nrest l)))] [else (nfilter p (nrest l))]))`
- $\boxed{1}$ `(cond [(nempty? l) nempty] [(p (nfirst l)) (ncons (nfirst l) (nfilter p (nrest l)))] [else (nfilter p (nrest l))])`
- $\boxed{2}$ `(nempty? l)`
- $\boxed{3}$ `nempty`
- $\boxed{4}$ `(p (nfirst l))`
- $\boxed{5}$ `(nfirst l)`
- $\boxed{6}$ `(ncons (nfirst l) (nfilter p (nrest l)))`
- $\boxed{7}$ `(nfirst l)`
- $\boxed{8}$ `(nfilter p (nrest l))`
- $\boxed{9}$ `(nrest l)`
- $\boxed{10}$ `else`
- $\boxed{11}$ `(nfilter p (nrest l))`
- $\boxed{12}$ `(nrest l)`

Luego, vamos a analizar el tipo de expresiones que encontramos.

- Para la cajita cero,
$$[[\boxed{0}]] = [[p]] \times [[l]] \to [[\boxed{1}]]$$
- Para la primer cajita,
$$\begin{aligned}[[\boxed{1}]] &= [[\texttt{(cond [}\boxed{2}\ \boxed{3}\texttt{] [}\boxed{4}\ \boxed{6}\texttt{] [}\boxed{10}\ \boxed{11}\texttt{])}]] \\ &= [[\boxed{2}]] \to [[\boxed{3}]]\texttt{or}[[\boxed{4}]] \to [[\boxed{6}]]\texttt{or}[[\boxed{10}]] \to [[\boxed{11}]]\end{aligned}$$
de donde
  - $[[\boxed{2}]] = \texttt{boolean}$
  - $[[\boxed{4}]] = \texttt{boolean}$
  - $[[\boxed{3}]] = [[\boxed{6}]] = [[\boxed{11}]]$
- Para la segunda cajita,
$$[[\boxed{2}]] = \texttt{(nempty? l)}$$
de donde
  - $[[$`(nempty? l)`$]] =$ `boolean`
  - $[[$`b`$]] =$ `nlist`

- Para la tercer cajita,

$$[[\boxed{3}]] = \texttt{nempty} = \texttt{nlist}$$

- Para la cuarta cajita,

$$\begin{aligned}[[\boxed{4}]] &= \texttt{(p (nfirst l))}\\ &= \texttt{(p }\boxed{5}\texttt{)}\end{aligned}$$

  donde $[[\texttt{p}]] = [[\boxed{5}]] \to [[\boxed{4}]]$
- Para la quinta cajita,

$$[[\boxed{5}]] = \texttt{(nfirst l)}$$

  de donde
  - $[[\texttt{((nfirst l))}]] = \texttt{number}$
  - $[[\texttt{l}]] = \texttt{nlist}$
- Para la sexta cajita,

$$\begin{aligned}[[\boxed{6}]] &= [[\texttt{(ncons (nfirst l) (nfilter p (nrest l)))}]]\\ &= [[\texttt{(ncons }\boxed{7}\ \boxed{8}\texttt{)}]]\end{aligned}$$

  de donde
  - $[[\boxed{6}]] = \texttt{nlist}$
  - $[[\boxed{7}]] = [[\boxed{5}]] = \texttt{number}$
  - $[[\boxed{8}]] = \texttt{(nfilter p (nrest l))} = \texttt{(nfilter p }\boxed{9}\texttt{)}$
    - $[[\texttt{nfilter}]] = [[\texttt{p}]][[\boxed{9}]] \to [[\boxed{8}]] = \texttt{nlist}$
- Para la novena cajta,

$$[[\boxed{9}]] = \texttt{(nrest l)}$$

  de donde
  - $[[\texttt{(nrest l)}]] = \texttt{nlist}$
  - $[[\texttt{l}]] = \texttt{nlist}$
- Para la décima cajita,

$$[[\boxed{10}]] = [[\texttt{else}]] = [[\texttt{true}]] = \texttt{boolean}$$

- Para la undécima cajita,

$$[[\boxed{11}]] = [[\boxed{8}]] = \texttt{nlist}$$

- Para la duodécima cajita,

$$[[\boxed{12}]] = [[\boxed{9}]] = \texttt{nlist}$$

Por lo tanto, el tipo de la función `nfilter` es

`nfilter: (number -> boolean) nlist -> nlist`

donde p es una función del tipo `(number -> boolean)` y l es del tipo `nlist`.

4. Usando el algoritmo de unificación, muestra la inferencia de tipos de las siguientes expresiones:

   - `(λ (x) (x 2 3))`
     Con $\boxed{1}(\boxed{2}\lambda(x)(\boxed{4}x\boxed{3}2\ 3))$:

Action	Stack	Substitution

Initialize	$[[\boxed{1}]] = [[\boxed{2}]] \to [[\boxed{3}]]$ $[[\boxed{2}]] = [[\boxed{4}]]$ $[[\boxed{4}]] = [[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{5}]] = \mathit{number}$ $[[\boxed{6}]] = \mathit{number}$	empty
step2	$[[\boxed{2}]] = [[\boxed{4}]]$ $[[\boxed{4}]] = [[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{5}]] = \mathit{number}$ $[[\boxed{6}]] = \mathit{number}$	$[[\boxed{1}]] \to [[\boxed{2}]] \to [[\boxed{3}]]$
step2	$[[\boxed{4}]] = [[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{5}]] = \mathit{number}$ $[[\boxed{6}]] = \mathit{number}$	$[[\boxed{1}]] \to [[\boxed{2}]] \to [[\boxed{3}]]$ $[[\boxed{2}]] \to [[\boxed{4}]]$
step2	$[[\boxed{5}]] = \mathit{number}$ $[[\boxed{6}]] = \mathit{number}$	$[[\boxed{1}]] \to ([[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]) \to [[\boxed{3}]]$ $[[\boxed{2}]] \to [[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{4}]] \to [[\boxed{5}]] \times [[\boxed{6}]] \to [[\boxed{3}]]$
stape2	$[[\boxed{6}]] = \mathit{number}$	$[[\boxed{1}]] \to (\mathit{number} \times [[\boxed{6}]] \to [[\boxed{3}]]) \to [[\boxed{3}]]$ $[[\boxed{2}]] \to \mathit{number} \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{4}]] \to \mathit{number} \times [[\boxed{6}]] \to [[\boxed{3}]]$ $[[\boxed{5}]] \to \mathit{number}$
step2	empty	$[[\boxed{1}]] \to (\mathit{number} \times \mathit{number} \to [[\boxed{3}]]) \to [[\boxed{3}]]$ $[[\boxed{2}]] \to \mathit{number} \times \mathit{number} \to [[\boxed{3}]]$ $[[\boxed{4}]] \to \mathit{number} \times \mathit{number} \to [[\boxed{3}]]$ $[[\boxed{5}]] \to \mathit{number}$ $[[\boxed{6}]] \to \mathit{number}$

- `((λ (x) (* x 2)) (+ 2 3))`
  Con: $\boxed{1}((\boxed{2}\lambda(x)\ \boxed{4}(*\ x\ 2))\ \boxed{3}(+\ 2\ 3))$:

Action	Stack	Substitution
Initialize	$[[\boxed{2}]] = [[\boxed{3}]] \to [[\boxed{1}]]$ $[[\boxed{2}]] = [[x]] \to [[\boxed{4}]]$ $[[\boxed{3}]] = \mathit{number}$ $[[\boxed{4}]] = \mathit{number}$	empty
step2	$[[\boxed{3}]] \to [[\boxed{1}]] = [[x]] \to [[\boxed{4}]]$ $[[\boxed{3}]] = \mathit{number}$ $[[\boxed{4}]] = \mathit{number}$	$[[\boxed{2}]] \to [[\boxed{3}]] \to [[\boxed{1}]]$
step4	$[[\boxed{3}]] = [[x]]$ $[[\boxed{1}]] = [[\boxed{4}]]$ $[[\boxed{3}]] = \mathit{number}$ $[[\boxed{4}]] = \mathit{number}$	$[[\boxed{2}]] \to [[\boxed{3}]] \to [[\boxed{1}]]$
step2	$[[\boxed{1}]] = [[\boxed{4}]]$ $[[x]] = \mathit{number}$ $[[\boxed{4}]] = \mathit{number}$	$[[\boxed{2}]] \to [[x]] \to [[\boxed{1}]]$ $[[\boxed{3}]] \to [[x]]$
step2	$[[x]] = \mathit{number}$ $[[\boxed{4}]] = \mathit{number}$	$[[\boxed{2}]] \to [[x]] \to [[\boxed{4}]]$ $[[\boxed{3}]] \to [[x]]$ $[[\boxed{1}]] \to [[\boxed{4}]]$

step2	$[[\boxed{4}]] = number$	$[[\boxed{2}]] \rightarrow number \rightarrow [[\boxed{4}]]$ $[[\boxed{3}]] \rightarrow number$ $[[\boxed{1}]] \rightarrow [[\boxed{4}]]$ $[[x]] \rightarrow number$
step2	empty	$[[\boxed{2}]] \rightarrow number \rightarrow number$ $[[\boxed{3}]] \rightarrow number$ $[[\boxed{1}]] \rightarrow number$ $[[x]] \rightarrow number$ $[[\boxed{4}]] \rightarrow number$

5. Indica si el sistema de Macros de `RACKET` y `C` es *higiénico* o no. Justifica con un pequeño programa que haga uso de macros.
   El sistema de Macros de `RACKET` es *higiénico*:

```
(define x 31)

(define-macro (mac)
  #'(begin
       (define x 62)
       (println x)))

(mac) ;;  62
(println x);;  31
```

Esto debido a que, se define una variable x fuera de la macro mac; otra variable se define por dentro. El (println x) generado por la macro se refiere a la x definida por la macro. El (println x) fuera de la macro se refiere a la x definida fuera de la macro. Entonces terminamos con dos identificadores llamados x con diferentes valores (pues viven en contextos léxicos diferentes).
El sistema de Macros de `C` es *no higiénico*

```
#define INCI(i) do { int INCIa = 0; ++i; } while (0)
int main(void)
{
    int x = 5, y = 7;
    INCI(x);
    INCI(y);
    printf("x es ahora %d, y es ahora %d\n", x, y);
    return 0;
}

x es ahora 6, y es ahora 8
```

Esto debido a que, no terminamos con dos identificadores y siempre se refiere a la mismas variables x e y haciendo que estas cambien de valor cada que utilicemos la macro.