Lenguajes de Programación, 2021-1
Tarea 6

Karla Ramírez Pulido Manuel Soto Romero Alejandro Hernández Mora
Alma Rocío Sánchez Salgado Silvia Díaz Gómez

Fecha de inicio: 14 de enero de 2021
Fecha de entrega: 21 de enero de 2021

1. Evalúa el siguiente código en RACKET, explica su resultado, y da la continuación asociada a evaluar, usando la notación $\lambda\uparrow$.

```
> (define c #f)

> (+ 1 (+ 2 (+ 3 (+ (let/cc k (set! c k) 4) 5))))

> (c 10)
```

2. Modifica cada una de las siguientes funciones de forma que usen el estilo de paso de continuaciones (*Continuation Passing Style)*.

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

(b)
```
(define (suma-digitos n)
        (if (< n 10)
            n
            (+ (modulo n 10) (suma-digitos (quotient n 10)))))
```

(c)
```
(define (cuadrados lst)
        (if (empty? lst)
            empty
            (cons (expt (car lst) 2) (cuadrados xs))))
```

(d)
```
(define (reversa lst)
      (if (empty? lst)
          empty
          (append (reversa (cdr lst)) (list x))))
```

3. Evalúa la siguiente expresión usando el paso de parámetros que se indica.

```
{with {{a 8}
       {b -8}
       {swap {fun {x y}
                  {with {{tmp x}}
                     {seqn {set x y}
                           {set y tmp}}}}}}
   {seqn {swap a b}
         {- a {+ b a}}}}
```

(a) Paso de parámetros por valor.

(b) Paso de parámetros por referencia.

4. Evalúa la siguiente expresión usando el paso de parámetros que se indica.

```
{with {{a 1}
       {foo {fun {x}
                 {seqn {set a {+ {* x -2} a}}
                       a}}}}
   {{fun {y} {+ y {- y y}} {foo 3}}}
```

(a) Paso de parámetros por nombre.

(b) Paso de parámetros por necesidad.

5. Evalúa la siguiente expresión usando el paso de parámetros por referencia-regreso

```
{with {{b 2}
       {f {fun {x} {seqn {set x 4}
                         {+ x b}}}}}
   {+ {f b} b}}
```

6. Dada la siguiente del definición del predicado `primo?` que decide si un número positivo mayor a 2 es primo. Modifícala usando memoización.

```
(define (primo? n)
   (if (equal? n 2)
       #t
       (aux n 2 (sub1 n))))

(define (aux n i j)
   (cond
      [(> i j) #t]
      [(zero? (modulo n i)) #f]
      [else (aux n (add1 i) j)]))
```