Auxiliar Diez - CC41A
Lenguajes de Programación

Oscar E. A. Callaú
oalvarez@dcc.uchile.cl

`Santiago - Chile, 27/Oct/2008`

1. Implementando Colas

En el capítulo de Representación procedural, hemos visto como podemos abstraer la construcción de estructuras de datos. Consideremos el caso de las colas.

Un conjunto de interfaces para las colas puede incluir operaciones para restablecer la a su estado inicial (vacía), verificar si la cola esta vacía, encolar un valor o remover un elemento de la cola.

Si trabajamos con funciones, estas deben tener como parámetro a la misma cola donde se quiere trabajar y si hacemos modificaciones, deben retonar la. Sin embargo, es mejor mantener la cola de una manera compartida entre esos procedimientos, ya que es difícil pasar la cola cada vez que se tenga que trabajar con ella. En estas situaciones es mejor trabajar con procedimientos que referencien a la cola como un objeto compartido que puede mutar su contenido.

La representación de esta cola tiene que ser transparente y solo debe proveer un conjunto de operaciones mínimas para trabajar con ella. Estas operaciones son:

(create-queue) , que crea una cola, mas el conjunto de operaciones básicas que trabajarán sobre la cola compartida (transparente para los demás procedimientos).

(queue-get-reset-operation queue) , retorna una función que restablece los valores iniciales de la cola.

(queue-get-empty?-operation queue) , retorna una función que verifica si la cola esta vacía.

(queue-get-enqueue-operation queue) , retorna la operación de encolar en la cola.

(queue-get-dequeue-operation queue) , retorna la operación de decolar sobre la cola.

Se le solicita a Ud. implemente estas interfaces.
Hint: Vea la facilidad de usar dos colas, una `q-in` *y la otra* `q-out`.

2. Store

Considere el siguiente intérprete de un lenguaje con estructuras de datos mutables (box), escrito usando *store-passing style* :

```
(define (interp expr env store)
   (type-case Expr expr
     ...
     [if0 (test then else)
         (if (num-zero? (interp test env store))
             (interp then env store)
             (interp else env store))) ...))
```

1. Qu é problema tiene este intérprete?
2. Escriba un programa que ilustre que la sem ántica del `if0` es err ónea.

3. Variables Mutables

De las siguientes expresiones, justifique los resultados que obtienen al ser evaluadas:

- ```
  {with {init {fun {a}
                   {setbox a 0}}}
        {with {z 5}
              {with {x {box z}}
                    {seqn
                     {init x}
                     {set z 10}
                     {unbox x}}}}}
  ```

- ```
  {with {setter {refun {x}
                       {set x 41}}}
        {with {y 10}
              {with {a {newbox y}}
                    {seqn
                     {setter y}
                     {+ {openbox a} y}}}}}
  ```