Lenguajes de Programación *Juan Carlos Faz Leal*
Tarea 06: EXPLICIT-REFS
26 de octubre del 2022

1. Ejercicio 4.8 [*]: Muestre exactamente dónde en nuestra implementación del almacenamiento estas operaciones toman tiempo lineal en lugar de tiempo constante.

   Las operaciones que contiene nuestra implementación hasta el momento son las siguientes: newref, deref y setref!

```
newref : ExpVal → Ref
(define newref
    (lambda (val)
        (let ((next-ref (length the-store)))
        (set! the-store (append the-store (list val)))
        next-ref)))
```

   Como las operaciones que se aplican en newref son length y append, y a su vez estas toman un tiempo lineal, por lo tanto newref es de tiempo lineal.

```
deref : Ref → ExpVal
(define deref
    (lambda (ref)
        (list-ref the-store ref)))
```

   La operacion que hacemos es list-ref, como listar toma tiempo lineal entonces podemos decir que list-ref es de tiempo lineal, por lo tanto deref toma tiempo lineal.

```
setref! : Ref × ExpVal → Unspecified
usage: sets the-store to a state like the original, but with
    position ref containing val.
(define setref!
    (lambda (ref val)
        (set! the-store
            (letrec
                ((setref-inner
                usage: returns a list like store1, except that
                position ref1 contains val.
                (lambda (store1 ref1)
                    (cond
                        ((null? store1)
                            (report-invalid-reference ref the-store))
                        ((zero? ref1)
                            (cons val (cdr store1)))
                        (else
```

```
                    (cons
                        (car store1)
                        (setref-inner
                            (cdr store1) (- ref1 1))))))))
        (setref-inner the-store ref)))))
```

En este problema observamos que la operación aplicada en setref! es setref-inner. Vemos que se usa un let recursivo por tanto podemos decir que hay un recorrido o loop o ciclo. Por tanto sabemos que cuando se usan ciclos no anidados son de tiempo lineal, entonces setref! es de tiempo lineal.

2. Ejercicio 4.9 [*]: Implemente el almacenamiento en tiempo constante representándolo como un vector Scheme. ¿Qué se pierde al usar esta representación?

   En cuanto a las desventajas de usar un vector para implementar el almacenamiento, puede ser que el problema surga cuando el vector este completamente lleno. Varias soluciones existen, se podria crear un nuevo almacenamiento con el doble de longitud y guardar el almacenamiento anterior en el nuevo.