Lenguaje Let

Kevin Ruiz

1. Sintáxis concreta

Expression := Number
Expression := -(Expression, Expression)
Expression := zero? (Expression)
Expression := if Expression then Expression else Expression
Expression := Identifier
Expression := let Identifier = Expression in Expression

2. Sintáxis abstracta

(const-exp num)
(diff-exp exp1 exp2)
(zero?-exp exp1)
(if-exp exp1 exp2 exp3)
(var-exp var)
(let-exp var exp1 body)

3. Semántica

```
(value-of (const-exp n) env) = (num-val n)

    (value-of (var-exp var) env) = env(var)

    (value-of (diff-exp exp1 exp2) env)
      = (num-val (- (expval->num (value-of exp1 env))
                    (expval->num (value-of exp2 env))))

    (value-of (zero?-exp exp1) env)
      = (let ([val1 (value-of exp1 env)])
           (bool-val (= 0 (expval->num val1))))

    (value-of (if-exp exp1 exp2 exp3) env)
      = (if (expval->bool (value-of exp1 env))
            (value-of exp2 env)
            (value-of exp3 env))

    (value-of (let-exp var exp1 body) env)
      = (let ([val1 (value-of exp1 env)])
           (value-of body [var = val1]env))
```