CS314 Homework 7

Sample solution

Spring 2014

1. (a) `((lambda (x)((lambda (y) ((lambda (z) e) v3)) v2)) v1)`

   (b)
```
(define maxSquareVal
  (lambda (l)
    (let ((sqrl (map (lambda (x) (* x x)) l)))
      (reduce
       (lambda (x y) (if (> x y) x y))
       sqrl
       0))))
```

2. Aredex for a β -reduction step has the general form `((lambda(x).M) N)`. In this problem, each redex is marked by a box, i.e., by $\boxed{\text{((lambda(x).M) N)}}$.

   (a) ($\boxed{((\lambda x.x)\ (\lambda x.1))}$ $(\lambda y.y)$)
   $\boxed{((\lambda x.1)\ (\lambda y.y))}$
   1

   (b) $\boxed{((\lambda x.((\lambda z.((\lambda x.(z\ x))\ 3))\ (\lambda y.(+\ x\ y))))\ 1)}$
   $\boxed{((\lambda z.((\lambda x.(z\ x))\ 3))\ (\lambda y.(+\ 1\ y)))}$
   $\boxed{((\lambda x.((\lambda y.(+\ 1\ y))\ x))\ 3)}$
   $\boxed{((\lambda y.(+\ 1\ y))\ 3)}$
   $\boxed{(+\ 1\ 3)}$
   4 (using a magic β-reduction)

   (c) $\boxed{((\lambda z.((\lambda y.z)\ ((\lambda x.(x\ x))(\lambda x.(x\ x)))))\ 5)}$
   $\boxed{((\lambda y.5)\ ((\lambda x.(x\ x))(\lambda x.(x\ x))))}$
   5

   Yes, the order makes a difference. If we applied (λx.(x x)) to (λx.(x x)) the result would again be ((λx.(x x))(λx.(x x))) So, if you choose to always pick the ((λx.(x x))(λx.(x x))) redex, you will never reach a normal form.

3. (a) TRUE: λa.λb.a
   FALSE: λa.λb.b
   OR: λx.λy.((x true) y)

   ((or true) false)
   (((λx.λy.((x true) y)) true) false) $\Rightarrow_\beta$ (λy.((true true) y) false)
   (λy.((true true) y) false) $\Rightarrow_\beta$ ((true true) false)
   (((λa.λb.a) true) false) $\Rightarrow_\beta$ ((λb.true) false)
   ((λb.true) false) $\Rightarrow_\beta$ true

(b) AND: λx.λy.((x y) false)

Proof that the above definition implements logical "AND"; to do this, we need to show that `((and x) y)` has the following property:

((and true) false) $\Rightarrow_\beta$ false
((and true) true) $\Rightarrow_\beta$ true
((and false) true) $\Rightarrow_\beta$ false
((and false) false) $\Rightarrow_\beta$ false

<u>case: assume x is `true`</u>: This means ((true y) false), which evalutes to `y`. So if x is true, then the expression reduces to `true` if y is `true`, or to `false`, if y is `false`.
<u>case: assume x is `false`</u>: This means ((false y) false), which always returns `false` no matter what the value of y is.
Therefore, the given implementation of "AND" is correct.

(c) NOT: λx.((x false) true)
XOR: λx.λy.((x (NOT y)) y)

Proof that the above definition implements logical "XOR"; to do this, we need to show that `((xor x) y)` has the following property:

((xor true) false) $\Rightarrow_\beta$
((true (NOT false)) false) $\Rightarrow_\beta$
((true true) false) $\Rightarrow_\beta$
((λb.true) false) $\Rightarrow_\beta$
true

((xor true) true) $\Rightarrow_\beta$
((true (NOT true)) true) $\Rightarrow_\beta$
((true false) true) $\Rightarrow_\beta$
((λb.false) true) $\Rightarrow_\beta$
false

((xor false) true) $\Rightarrow_\beta$
((false (NOT false)) true) $\Rightarrow_\beta$
((false false) true) $\Rightarrow_\beta$
((λb.b) true) $\Rightarrow_\beta$
true

((xor false) false) $\Rightarrow_\beta$
((false (NOT false)) false) $\Rightarrow_\beta$
((false true) false) $\Rightarrow_\beta$
((λb.b) false) $\Rightarrow_\beta$
false

Therefore, the given implementation of "XOR" is correct.

4. $K \equiv \lambda xy.x$
$S \equiv \lambda xyz.((xz)(yz))$
$I \equiv \lambda x.x$

$((S\ K)\ K) \Rightarrow_\beta$
$(((\lambda xyz.((xz)(yz)))\ K)\ K) \Rightarrow_\beta$
$((\lambda yz.((K\ z)(yz)))\ K) \Rightarrow_\beta$
$(\lambda z.((K\ z)(K\ z))) \Rightarrow_\beta$
$(\lambda z.(((\lambda xy.x)\ z)(K\ z))) \Rightarrow_\beta$
$(\lambda z.((\lambda y.z)(K\ z)) \Rightarrow_\beta$
$(\lambda z.z) \Rightarrow_\alpha$
$(\lambda x.x)$ Note that $(\lambda z.z) = (\lambda x.x)$ after renaming