Midterm Examination
CS 565
March 11, 2009

Name: ____

Instructions: Answer all questions in the space provided. Extra blank pages are provided in the back. Partial credit will be given where appropriate.

Maximum score: 100
Attained score: ____________________________

Question 1. (10 points)

Let $A = \lambda a \lambda b \lambda c \lambda d \lambda e \lambda f \lambda g \lambda h \lambda i \lambda j \lambda k \lambda l \lambda m \lambda n \lambda o \lambda p \lambda q \lambda s \lambda t \lambda u \lambda v \lambda w \lambda x \lambda y \lambda z \lambda r . r(isthisafixedpointcombinator)$ and let $B = AAAAAAAAAAAAAAAAAAAAAAAAAA$ (26 applications of A).

Is B a fixed-point combinator? Justify your answer.

Yes. If B is a fixed-point combinator, then $B\ A = A(B\ A)$. $B\ A = A[26 \text{ applications of } A]A = A\ (B\ A)$.

Question 2. (20 points)

Let $f \circ g = \lambda x.f(g\ x)$ for all f, g (this is simply function composition). Let $\Delta = \lambda x.(x\ x)$.

1. (5 points) Show that $\Delta(f \circ \Delta) \equiv f(\Delta(f \circ\ \Delta))$

   $\Delta(f \circ\ \Delta)$
   $=\ (\lambda\ x.(x\ x))\ (\lambda\ x.f(\Delta\ x))$
   $=\ (\lambda\ x.f(\Delta\ x))(\lambda\ x.f(\Delta\ x))$
   $=\ f(\Delta\ (\lambda\ x.f(\Delta\ x)))$
   $=\ f(\Delta\ (f \circ \Delta))$

2. Let $Y_1 = \lambda f.\Delta(f \circ \Delta)$, $G = \lambda y.\lambda f.f(y\ f)$, and $Y_{n+1} = Y_n G$. Prove:

   (a) (5 points) For all $x, Y_1\ x \equiv\ x(Y_1\ x)$

   $Y_1\ x = \Delta(x \circ\ \Delta)$
   $=\ (\lambda\ x.(x\ x))\ (x \circ\ \Delta)$
   $=\ (x \circ\ \Delta)\ (x \circ\ \Delta)$
   $=\ (\lambda\ z.x(\Delta\ z))\ (x \circ\ \Delta)$
   $=\ (x(\Delta\ (x \circ\ \Delta)))$
   $= x(Y_1\ x)$

   (b) (5 points) $G\ Y_1 \equiv\ Y_1$

   $G\ Y_1 = \lambda\ f.f(Y_1\ f)$
   $\Delta(f \circ \Delta)$
   $= (\lambda\ x.(x\ x))\ (f \circ \Delta)$
   $= (f \circ\ \Delta)(f \circ\ \Delta)$
   $= (\lambda\ x.f(\Delta(x))$
   $= f(\Delta(f \circ\ \Delta))$
   $= f(Y_1\ f)$

(c) (5 points) Show that all members of the sequence $Y_1, \ldots$ are fixed point combinators.

By induction over n. The base case is trivial. Now, $GY^{n+1} = G(Y^n G)$
$= Y^n G$ since Y^n is a fixed-point combinator by IH.
$\equiv Y^{n+1}$

Question 3. (10 points)

Using the axiom of assignment and the rule of composition prove the following sequence of statements exchanges the values of variables x and y.

$$\begin{array}{rcl} x & := & x - y; \\ y & := & x + y; \\ x & := & y - x \end{array}$$

Make sure to clearly state appropriate pre- and post-conditions in your proof.

The precondition is $\{x = A \land y = B\}$. The postcondition is $\{x = B \land y = A\}$. Working backwards, we have:

$$\{P_1\}x := y - x\{x = B \land y = A\}$$

Using the axiom of assignment, we get:

$$\{y - x = B \land y = A\}x := y - x\{x = B \land y = A\}$$

We apply a similar process using the newly derived precondition as the last the postcondition for the second-last statement:

$$\{x + y - x = B \land x + y = A\}y := x + y\{y - x = B \land y = A\}$$

We simplify the precondition to $\{y = B \land x + y = A\}$. Applying the rule of sequencing and repeating our use of the axiom of assignment, we get:

$$\{y = B \land x - y + y = A\}x := x - y\{y = B \land x + y = A\}$$

The precondtion can be simplified to $\{y = B \land x = A\}$. We apply the rule of composition once more to prove the validity of the entire program.

Question 4. (15 points)

The **IMP** language is given by the following grammar:

$$\begin{array}{lll} e & \in \textbf{AExp} ::= & n \,|\, x \,|\, e_1 + e_2 \,|\, e_1 - e_2 \,|\, e_1 * e_2 \\ b & \in \textbf{BExp} ::= & \textbf{true} \,|\, \textbf{false} \,|\, e_1 = e_2 \,|\, e_1 \leq e_2 \,|\, \neg b \,|\, b_1 \wedge b_2 \,|\, b_1 \vee b_2 \\ c & \in \textbf{Com} ::= & \textbf{skip} \,|\, x := e \,|\, c_1; c_2 \,|\, \textbf{if}\, b\, \textbf{then}\, c_1\, \textbf{else}\, c_2 \,|\, \textbf{while}\, b\, \textbf{do}\, c \end{array}$$

Prove the following statement by structural induction: For any boolean command b and any initial state σ, such that $\sigma(x)$ is even, if

$$\textbf{while}\, b\, \textbf{do}\, x := x + 2, \sigma \Downarrow \sigma'$$

than $\sigma'(x)$ is even.

Reference: The natural semantics for **while** loops is given by the following two commands:

$$\frac{(b, \sigma) \Downarrow \textbf{false}}{(\textbf{while}\, b\, \textbf{do}\, c, \sigma) \Downarrow \sigma}$$

$$\frac{(b, \sigma) \Downarrow \textbf{true}, (c;\, \textbf{while}\, b\, \textbf{do}\, c), \sigma \Downarrow \sigma'}{(\textbf{while}\, b\, \textbf{do}\, c, \sigma) \Downarrow \sigma'}$$

Consider a derivation $\mathcal{D}$ of this expression. There are two possible rules used at the top of $\mathcal{D}$:

1. **while** false $\Downarrow \sigma = \sigma$ and $\sigma(x)$ is even by the induction hypothesis.
2. **while** true. In evaluating the antecedents of the true case, we have $\langle x := x + 2, \sigma\rangle \Downarrow \sigma''$ and $\mathcal{D}_1 = \langle \textbf{while}\, b\, \textbf{do}, ..., \sigma'' \Downarrow \sigma'\rangle$. We know that $\sigma'' = \sigma[x \mapsto \sigma(x) + 2]$ and thus, $\sigma''(x)$ is even. Apply the induction hypothesis to the derivation rooted at $\mathcal{D}_1$ to complete the proof.

Question 5. (20 points)

(a) (10 points) For each of the following lambda terms, decide whether it is typable in the simply-typed lambda calculus. If a term is typable, give a type for it.

i. $\lambda x.\ \lambda\ y.x(x\ y)$

$(\sigma_1 \to \sigma_2) \to \sigma_1 \to \sigma_2$

ii. $\lambda x.\ \lambda\ y.(y\ x)y$

Not typable

iii. $\lambda x.\ \lambda\ y.y(x\ y)$

$((\sigma_1 \to \sigma_2) \to \sigma_1) \to (\sigma_1 \to \sigma_2) \to \sigma_2$

(b) (10 points) For each of the following type formulas, provide a *closed* lambda term (i.e., a term with no free variables) in the simply typed lambda calculus that has that type, if one exists.

i. $\sigma \to \sigma$

$\lambda\, x : \sigma.x$

ii. $(\sigma_1 \to \sigma_2) \to \sigma_3 \to (\sigma_1 \to \sigma_2)$

$\lambda\, x : \sigma_1 \to \sigma_2.\lambda\, y : \sigma_3.x$

iii. $(\sigma_1 \to \sigma_2) \to (\sigma_2 \to \sigma_3) \to (\sigma_1 \to \sigma_3)$

$\lambda\, f.\sigma_1 \to \sigma_2.\lambda\, g : \sigma_2 \to \sigma_3.\lambda\, x : \sigma_1.g(fx)$

Question 6. (10 points)

The **SKI**-calculus is comprised of three combinators:

$$\begin{aligned}\mathbf{S} &= \lambda\, x.\lambda\, y.\lambda\, z.x\ z\ (y\ z)\\ \mathbf{K} &= \lambda\, x.\lambda\, y.x\\ \mathbf{I} &= \lambda\, x.x\end{aligned}$$

1. (5 points) What is the λ-calculus term that corresponds to **S I I (S I I)**?

   $(\lambda\ x.(x\ x))(\lambda\ x.(x\ x))$

2. (5 points) What is the value of **S (K S) K** when reduced to normal form? Can you think of a good name for this term?

   This combinator reduces to $\lambda\, x.\, \lambda\, y.\lambda z.x\ (y\ z)$. It could be regarded as a function composition operation.

Question 7. (15 points)

Prove the following lemma: If $\Gamma \vdash e : \sigma$ then every free variable of e appears in Γ.

If you choose to use induction, you must clearly state the method of induction chosen, the base case, the induction hypothesis, the induction steps, and when you rely on the hypothesis.

By induction on typing derivations.

Base case.. By assumption $\Gamma \vdash x : \sigma$. Since any proof ends with $x : \sigma \vdash x : \sigma$ and x is the only free variable, the Lemma holds trivially.

Induction steps:

1. (Var): Suppose $\Gamma, x : \tau : e : \sigma$ follows from $\Gamma \vdash e : \sigma$. By IH, $FV(e) \in \Gamma$. Clearly, $FV(e) \in \Gamma \cup \{x : \tau\}$.
2. (Intro): Suppose $\Gamma \vdash \lambda\, x : \tau.e : (\tau \to \sigma)$ follows from $\Gamma, x : \tau \vdash e : \sigma$. By IH, $FV(e) \in \Gamma \cup \{x : \tau\}$. Since $FV(\lambda\, x : \tau.e) = FV(e) - \{x\}$, all free variables of $(\lambda\, x : \tau.e) \in \Gamma$.
3. (Elim): Suppose $\Gamma \vdash (e_1\ e_2)$ follows from $\Gamma \vdash e_1 : \sigma \to \tau$ and $\Gamma \vdash e_2 : \sigma$. By IH, $FV(e_1) \in \Gamma$ and $FV(e_2) \in \Gamma$. Thus, $FV(e_1\ e_2) = FV(e_1) \cup FV(e_2)$, and all $FV(e_1\ e_2) \in \Gamma$.