Name:____

CSE 505, Fall 2007, Midterm Examination
1 November 2007

Please do not turn the page until everyone is ready.

Rules:

- The exam is closed-book, closed-note, except for one side of one 8.5x11in piece of paper.
- **Please stop promptly at 11:50.**
- You can rip apart the pages, but please write your name on each page.
- There are **100 points** total, distributed **unevenly** among **4** questions (which have multiple parts).

Advice:

- Read questions carefully. Understand a question before you start writing.
- Write down thoughts and intermediate steps so you can get partial credit.
- The questions are not necessarily in order of difficulty. **Skip around.** In particular, make sure you get to all the problems.
- If you have questions, ask.
- Relax. You are here to learn.

Name:____

For your reference:

$$\begin{array}{rcl} s & ::= & \mathsf{skip} \mid x := e \mid s; s \mid \mathsf{if}\ e\ s\ s \mid \mathsf{while}\ e\ s \\ e & ::= & c \mid x \mid e + e \mid e * e \\ (c & \in & \{\ldots, -2, -1, 0, 1, 2, \ldots\}) \\ (x & \in & \{\mathtt{x}_1, \mathtt{x}_2, \ldots, \mathtt{y}_1, \mathtt{y}_2, \ldots, \mathtt{z}_1, \mathtt{z}_2, \ldots, \ldots\}) \end{array}$$

$\boxed{H\ ;\ e\ \Downarrow\ c}$

$$\text{CONST}\ \frac{}{H\ ;\ c\ \Downarrow\ c} \qquad \text{VAR}\ \frac{}{H\ ;\ x\ \Downarrow\ H(x)} \qquad \text{ADD}\ \frac{H\ ;\ e_1\ \Downarrow\ c_1 \qquad H\ ;\ e_2\ \Downarrow\ c_2}{H\ ;\ e_1 + e_2\ \Downarrow\ c_1 + c_2} \qquad \text{MULT}\ \frac{H\ ;\ e_1\ \Downarrow\ c_1 \qquad H\ ;\ e_2\ \Downarrow\ c_2}{H\ ;\ e_1 * e_2\ \Downarrow\ c_1 * c_2}$$

$\boxed{H_1\ ;\ s_1\ \rightarrow\ H_2\ ;\ s_2}$

$$\text{ASSIGN}\ \frac{H\ ;\ e\ \Downarrow\ c}{H\ ;\ x := e\ \rightarrow\ H, x \mapsto c\ ;\ \mathsf{skip}} \qquad \text{SEQ1}\ \frac{}{H\ ;\ \mathsf{skip}; s\ \rightarrow\ H\ ;\ s} \qquad \text{SEQ2}\ \frac{H\ ;\ s_1\ \rightarrow\ H'\ ;\ s_1'}{H\ ;\ s_1; s_2\ \rightarrow\ H'\ ;\ s_1'; s_2}$$

$$\text{IF1}\ \frac{H\ ;\ e\ \Downarrow\ c \qquad c > 0}{H\ ;\ \mathsf{if}\ e\ s_1\ s_2\ \rightarrow\ H\ ;\ s_1} \qquad \text{IF2}\ \frac{H\ ;\ e\ \Downarrow\ c \qquad c \leq 0}{H\ ;\ \mathsf{if}\ e\ s_1\ s_2\ \rightarrow\ H\ ;\ s_2} \qquad \text{WHILE}\ \frac{}{H\ ;\ \mathsf{while}\ e\ s\ \rightarrow\ H\ ;\ \mathsf{if}\ e\ (s; \mathsf{while}\ e\ s)\ \mathsf{skip}}$$

$$\begin{array}{rcl} e & ::= & \lambda x.\ e \mid x \mid e\ e \mid c \\ v & ::= & \lambda x.\ e \mid c \\ \tau & ::= & \mathsf{int} \mid \tau \rightarrow \tau \end{array}$$

$\boxed{e \rightarrow e'}$

$$\frac{}{(\lambda x.\ e)\ v \rightarrow e[v/x]} \qquad \frac{e_1 \rightarrow e_1'}{e_1\ e_2 \rightarrow e_1'\ e_2} \qquad \frac{e_2 \rightarrow e_2'}{v\ e_2 \rightarrow v\ e_2'}$$

$\boxed{e[e'/x] = e''}$

$$\frac{}{x[e/x] = e} \qquad \frac{e_1[e/x] = e_1' \qquad y \neq x \qquad y \notin FV(e)}{(\lambda y.\ e_1)[e/x] = \lambda y.\ e_1'}$$

$$\frac{y \neq x}{y[e/x] = y} \qquad \frac{e_1[e/x] = e_1' \qquad e_2[e/x] = e_2'}{(e_1\ e_2)[e/x] = e_1'\ e_2'}$$

$\boxed{\Gamma \vdash e : \tau}$

$$\frac{}{\Gamma \vdash c : \mathsf{int}} \qquad \frac{}{\Gamma \vdash x : \Gamma(x)} \qquad \frac{\Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x.\ e : \tau_1 \rightarrow \tau_2} \qquad \frac{\Gamma \vdash e_1 : \tau_2 \rightarrow \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau_1}$$

- If $\cdot \vdash e : \tau$ and $e \rightarrow e'$, then $\cdot \vdash e' : \tau$.
- If $\cdot \vdash e : \tau$, then e is a value or there exists an e' such that $e \rightarrow e'$.
- If $\Gamma, x{:}\tau' \vdash e : \tau$ and $\Gamma \vdash e' : \tau'$, then $\Gamma \vdash e[e'/x] : \tau$.

Name:____

1. In this problem, we consider an expression language that is like expressions in IMP except we remove multiplication and we add a *global counter*. Our syntax is:

$$e ::= c \mid x \mid e + e \mid \mathsf{next}$$

Informally, the next expression evaluates to the current counter-value and has the side-effect of incrementing the counter value.

   (a) (**11** points) Give a large-step semantics for this expression language. The judgment should have the form $H; c_1; e \Downarrow c_2; c$ where:

   - H, e, and c are like in IMP.
   - c_1 is the value of the global counter before evaluation.
   - c_2 is the value of the global counter after evaluation.

   (b) (**16** points) Prove this theorem: If $H; c_1; e \Downarrow c_2; c$ and $c_1' > c_1$, then there exist c_2' and c' such that $H; c_1'; e \Downarrow c_2'; c'$ and $c_2' > c_2$.

   (c) (**7** points) Suppose we also extend IMP statement semantics to support the global counter (so the judgment has the form $H; c; s \to H'; c'; s'$). Argue that this theorem is *false*: If $H_1; c_1; s \to^* H_2; c_2; \mathsf{skip}$ and $c_1' > c_1$, then there exist H_2' and c_2' such that $H; c_1'; s \to^* H_2'; c_2'; \mathsf{skip}$ and $c_2' > c_2$. *You do not need to give the semantic rules for statements or show a full state sequence. Just give an example showing the theorem is false and explain why informally.*

Name:____

(This page intentionally blank)

Name:____

2. (**10** points) In this problem we extend IMP statements with the construct $\mathsf{repeat}\ c\ s$. Informally, the idea is to execute s c times. Here are two *separate* ways one might add rules to the semantics:

   - First way:

$$\frac{c > 0}{H; \mathsf{repeat}\ c\ s \to H; (s; \mathsf{repeat}\ (c-1)\ s)} \qquad\qquad \frac{c \leq 0}{H; \mathsf{repeat}\ c\ s \to H; \mathsf{skip}}$$

   - Second way:

$$\frac{}{H; \mathsf{repeat}\ c\ s \to H; (s; \mathsf{if}\ (c-1)\ (\mathsf{repeat}\ (c-1)\ s)\ \mathsf{skip})}$$

   One of these ways is *wrong* (in some situations) according to the informal description.

   (a) Which way is wrong? Explain why it is wrong.

   (b) Show how to change the wrong way to make it correct.

Name:____

3. (**18** points) Note there is a part (a) and part (b) to this problem.

   (a) For each Caml function below (`q1`, `q2`, and `q3`):

   - Describe in 1–2 English sentences what the function computes.
   - Give the type of the function. (Hint: For all three functions, the type has one type variable.)

```
let q1 x =
  let rec g x y =
    match x with
      [] -> y
    | hd::tl -> g tl (hd::y)
  in g x []

let rec q2 f lst =
  match lst with
    [] -> []
  | hd::tl -> if f hd then hd::(q2 f tl) else q2 f tl

let q3 x g = g (g x)
```

   (b) Consider this purposely complicated code that uses `q3` as defined above.

```
let x = q3 2
let y z = z+z
let z = 9
let x = x y
```

   After evaluating this code, what is `x` bound to?

Name:____

4. In this problem, we consider a call-by-value lambda-calculus with very basic support for profiling: In addition to computing a value, it computes how many times an expression of the form $\mathsf{count}\ e$ is evaluated. Here is the syntax and operational semantics:

$$e ::= \lambda x.\ e \mid x \mid e\ e \mid c \mid \mathsf{count}\ e$$

$\boxed{c; e \rightarrow c'; e'}$

$$\frac{}{c; (\lambda x.\ e)\ v \rightarrow c; e[v/x]} \qquad \frac{c; e_1 \rightarrow c'; e_1'}{c; e_1\ e_2 \rightarrow c'; e_1'\ e_2} \qquad \frac{c; e_2 \rightarrow c'; e_2'}{c; v\ e_2 \rightarrow c'; v\ e_2'}$$

$$\frac{}{c; \mathsf{count}\ v \rightarrow c+1; v} \qquad \frac{c; e \rightarrow c'; e'}{c; \mathsf{count}\ e \rightarrow c'; \mathsf{count}\ e'}$$

Given a source program e, our initial state is $0; e$ (i.e., the count starts at 0). A program state $c; e$ type-checks if e type-checks (i.e., the count can be anything).

(a) (**6** points) Give a typing rule for $\mathsf{count}\ e$ that is sound and not unnecessarily restrictive.

(b) (**13** points) State an appropriate Preservation Lemma for this language. Prove just the case(s) directly involving $\mathsf{count}\ e$ expressions.

(c) (**13** points) State an appropriate Progress Lemma for this language. Prove just the case(s) directly involving $\mathsf{count}\ e$ expressions.

(d) (**6** points) Give an example program that terminates in our language *and* would terminate if we changed function application to be call-by-name *but* under call-by-name it would produce a different resulting count. (Hint: This should not be difficult.)

Name:____

(This page intentionally blank)