Sample Mid-Term Exam 2

CS 5510, Fall 2013

November 13

Name: ______________________________

Instructions: You have eighty minutes to complete this open-book, open-note, closed-interpreter exam. Please write all answers in the provided space, plus the back of the exam if necessary. provided space, plus the back of the exam if necessary.

1) Which of the following produce different results in a eager language and a lazy language? Both produce the same result if they both produce the same number or they both produce a procedure (even if the procedure doesn't behave exactly the same when applied), but they can differ in errors reported.

a) `{{lambda {y} 12} {1 2}}`

b) `{lambda {x} {{lambda {y} 12} {1 2}}}`

c) `{+ 1 {lambda {y} 12}}`

d) `{+ 1 {{lambda {x} {+ 1 13}} {+ 1 {lambda {z} 12}}}}`

e) `{+ 1 {{lambda {x} {+ x 13}} {+ 1 {lambda {z} 12}}}}`

2) Given the type rules

$$[\ldots \mathbf{x} \leftarrow \tau \ldots] \vdash \mathbf{x} : \tau \qquad \Gamma \vdash 1 : \texttt{num} \qquad \frac{\Gamma \vdash \mathbf{e}_1 : \texttt{num} \quad \Gamma \vdash \mathbf{e}_2 : \texttt{num}}{\Gamma \vdash \{\texttt{+}\ \mathbf{e}_1\ \mathbf{e}_2\} : \texttt{num}}$$

$$\frac{\Gamma[\mathbf{x} \leftarrow \tau_1] \vdash \mathbf{e} : \tau_2}{\Gamma \vdash \{\texttt{lambda}\ \{[\mathbf{x}\ \texttt{:}\ \tau_1]\}\ \mathbf{e}\} : (\tau_1 \rightarrow \tau_2)} \quad \frac{\Gamma \vdash \mathbf{e}_1 : (\tau_1 \rightarrow \tau_2) \quad \Gamma \vdash \mathbf{e}_2 : \tau_1}{\Gamma \vdash \{\mathbf{e}_1\ \mathbf{e}_2\} : \tau_2}$$

in one of the following expressions, the `____` can be filled in with a type so that the resulting expression has a type in the enmpty environment, while there is no type for the `____` that causes the other to have a type. Pick the right expression and show a derivation tree (which is a trace of `typecheck` that's written in the style as the type rules above) demonstrating that the chosen expression has a type.

```
{{lambda {[x : ____]} {+ x 1}} x}
```

```
{lambda {[x : ____]} {+ {x 1} 1}}
```

Note that your answer should not include symbols like Γ, τ, or $\mathbf{e}$, except when used as designated abbreviations, since those are meta-variables that are replaced by concrete environments, types, and expressions in the derivation tree.

3) Given the following expression:

```
{{lambda {x} {x x}}
 {lambda {y} 12}}
```

Describe a trace of the evalaution in terms of arguments to `interp` and `continue` functions for every call of each in the `lambda-k.rkt` interpreter. (There will be 7 calls to `interp` and 5 calls to `continue`.) The `interp` function takes three arguments — an expression, an environment, and a continuation — so show all three for each `interp` call. The `continue` function takes two arguments — a continuation and a value — so show both for each `continue` call. Represent continuations using records.

Answers

1) *a* and *d*.

2)

$$\dfrac{\dfrac{\dfrac{\Gamma_1 \vdash \texttt{x} : (\texttt{num} \to \texttt{num}) \quad \Gamma_1 \vdash \texttt{1} : \texttt{num}}{\Gamma_1 \vdash \{\texttt{x 1}\} : \texttt{num}} \qquad \Gamma_1 \vdash \texttt{1} : \texttt{num}}{\Gamma_1 = [\texttt{x} \leftarrow (\texttt{num} \to \texttt{num})] \vdash \{\texttt{+ } \{\texttt{x 1}\}\texttt{ 1}\} : \texttt{num}}}{\emptyset \vdash \{\texttt{lambda } \{[\texttt{x : } (\texttt{num} \to \texttt{num})]\}\ \{\texttt{+ } \{\texttt{x 1}\}\texttt{ 1}\}\} : ((\texttt{num} \to \texttt{num}) \to \texttt{num})}$$

3)

interp	expr	=	$\boxed{\texttt{\{\{lambda \{x\} \{x x\}\} \{lambda \{y\} 12\}\}}}$
	env	=	`mt-env`
	k	=	`(doneK)`
interp	expr	=	$\boxed{\texttt{\{lambda \{x\} \{x x\}\}}}$
	env	=	`mt-env`
	k	=	`(appArgK` $\boxed{\texttt{\{lambda \{y\} 12\}}}$ `mt-env (doneK))`
cont	k	=	`(appArgK` $\boxed{\texttt{\{lambda \{y\} 12\}}}$ `mt-env (doneK))`
	val	=	`(closV 'x` $\boxed{\texttt{\{x x\}}}$ `mt-env)` $= v_1$
interp	expr	=	$\boxed{\texttt{\{lambda \{y\} 12\}}}$
	env	=	`mt-env`
	k	=	`(doAppK` v_1 `(doneK))`
cont	k	=	`(doAppK` v_1 `(doneK))`
	val	=	`(closV 'y` $\boxed{\texttt{12}}$ `mt-env)` $= v_2$
interp	expr	=	$\boxed{\texttt{\{x x\}}}$
	env	=	`(extend-env (bind 'x` v_2`) mt-env)` $= e_1$
	k	=	`(doneK)`
interp	expr	=	$\boxed{\texttt{x}}$
	env	=	e_1
	k	=	`(appArgk` $\boxed{\texttt{x}}$ e_1 `(doneK))`
cont	k	=	`(appArgK` $\boxed{\texttt{x}}$ e_1 `(doneK))`
	val	=	v_2
interp	expr	=	$\boxed{\texttt{x}}$
	env	=	e_1
	k	=	`(doAppK` v_2 `(doneK))`
cont	k	=	`(doAppK` v_2 `(doneK))`
	val	=	v_2
interp	expr	=	$\boxed{\texttt{12}}$

	env	=	`(extend-env (bind 'y` v_2`) mt-env)`
	k	=	`(doneK)`
cont	k	=	`(doneK)`
	val	=	`(numV 12)`