Sample Mid-Term Exam 2 (take-home)

CS 6510, Spring 2016

actual exam due March 30

Name: ________________________________

Start time: ____________________
End time: ____________________

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

1) Which of the following produce different results in an 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 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. (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 value and a continuation — so show both for each `continue` call. Represent continuations using records.

3) Suppose a garbage-collected interepreter uses the following three kinds of records:

- Tag **1**: a record containing two pointers
- Tag **2**: a record containing one pointer and one integer
- Tag **3**: a record containing one integer

The interpreter has one register, which always contains a pointer, and a memory pool of size 22. The allocator/collector is a two-space copying collector, so each space is of size 11. Records are allocated consecutively in to-space, starting from the first memory location, 0.

The following is a snapshot of memory just before a collection where all memory has been allocated:

- Register: 8
- To space: 1 3 8 3 0 2 3 7 2 0 8

What are the values in the register and the new to-space (which is also addressed starting from 0) after collection? Assume that unallocated memory in to-space contains 0.

- Register:
- To space:

Answers

1) *a* and *d*.

2)

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	val	=	`(closureV 'x` $\boxed{\texttt{\{x x\}}}$ `mt-env)` $= V_1$
	k	=	`(appArgK` $\boxed{\texttt{\{lambda \{y\} 12\}}}$ `mt-env (doneK))`
interp	expr	=	$\boxed{\texttt{\{lambda \{y\} 12\}}}$
	env	=	`mt-env`
	k	=	`(doAppK` V_1 `(doneK))`
cont	val	=	`(closureV 'y` $\boxed{\texttt{12}}$ `mt-env)` $= V_2$
	k	=	`(doAppK` V_1 `(doneK))`
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	val	=	V_2
	k	=	`(appArgK` $\boxed{\texttt{x}}$ E_1 `(doneK))`
interp	expr	=	$\boxed{\texttt{x}}$
	env	=	E_1
	k	=	`(doAppK` V_2 `(doneK))`
cont	val	=	V_2
	k	=	`(doAppK` V_2 `(doneK))`
interp	expr	=	$\boxed{\texttt{12}}$
	env	=	`(extend-env (bind 'y` V_2`) mt-env)`
	k	=	`(doneK)`
cont	val	=	`(numV 12)`
	k	=	`(doneK)`

3) Register: 0, To space: 2 3 8 1 6 0 3 0 0 0 0