Massachusetts Institute of Technology

6.035 Computer Language Engineering

Fall 2006

Quiz 2

Tuesday, November 7th, 2006

Name: ____

*Athena username: ________________________________

This quiz is open book, open notes. You have 50 minutes to complete it. It contains 14 questions in 10 pages (including this one), totaling 100 points. Page 11 can be detached and used as a reference for questions 3 to 9. Before you start, please check your copy to make sure it is complete.

Write your initials and section number on the top of ALL pages.

Please write neatly; we cannot give credit for what we cannot read.

Good luck!

1			of 5
2			of 9
3			of 4
4			of 8
5			of 8
6			of 8
7			of 11
8			of 6
9			of 8
10			of 5
11			of 10
12			of 6
13			of 6
14			of 6
			of 100

*Athena is MIT's UNIX-based computing environment. OCW does not provide access to it.

Question 1. Register saving 5 points

In the following program snippet:

```
void main() { □
 □   int A, B, C, D, E; □
     … □
 □   A = B + C; □
 □   E = A + C; □
 □   foo( ); □
 □   E = A + B; □     □
} □
```

No optimizations are performed and nothing is known about `foo()`, to minimize saves and restores of the registers, in what type of register should we allocate each variable?
(circle the corresponding register type for each variable)

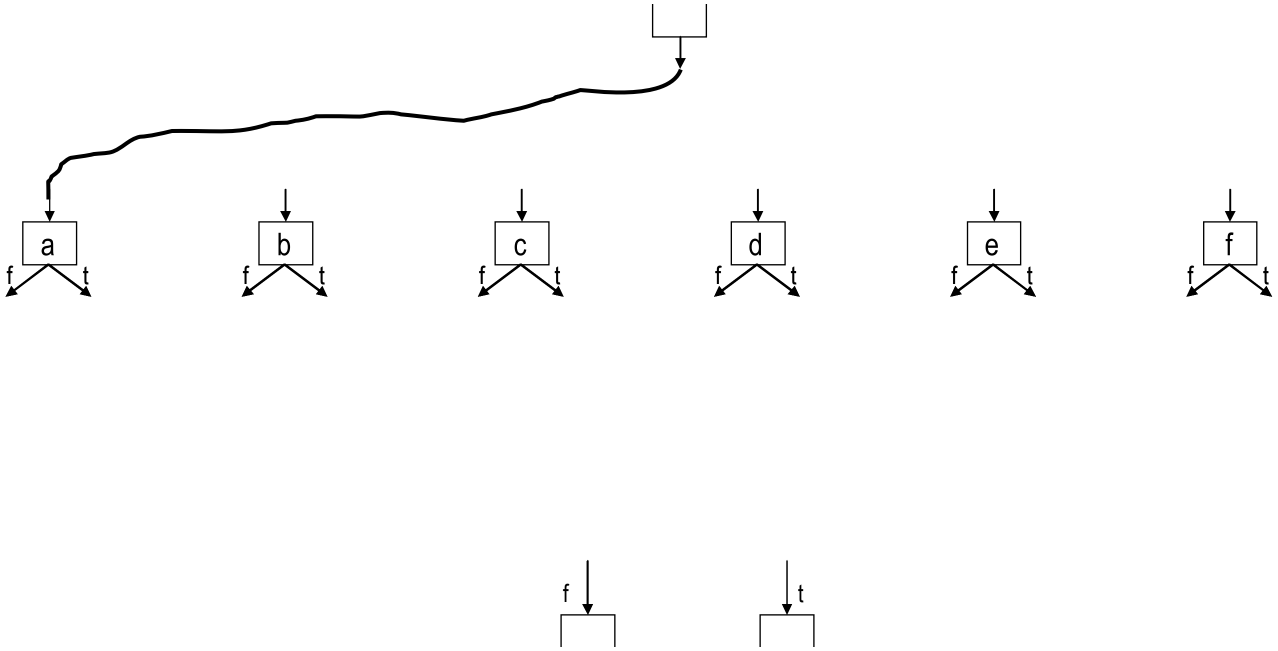
(a)	Variable A	in a caller saved register	in a callee Saved register	in any register	don't allocate a register
(b)	Variable B	in a caller saved register	in a callee Saved register	in any register	don't allocate a register
(c)	Variable C	in a caller saved register	in a callee Saved register	in any register	don't allocate a register
(d)	Variable D	in a caller saved register	in a callee Saved register	in any register	don't allocate a register
(e)	Variable E	in a caller saved register	in a callee Saved register	in any register	don't allocate a register

Question 2. Short-Circuiting 9 points

In evaluating the following expression

$(a \wedge b) \vee (c \wedge (d \vee e \vee f))$

What is the proper short-circuited evaluation?
(connect the control flow arrows from the input block through testing of all the variables to the two output blocks to form the proper short-circuited evaluation. Note that the line from the input block to ***a*** *is shown.)*



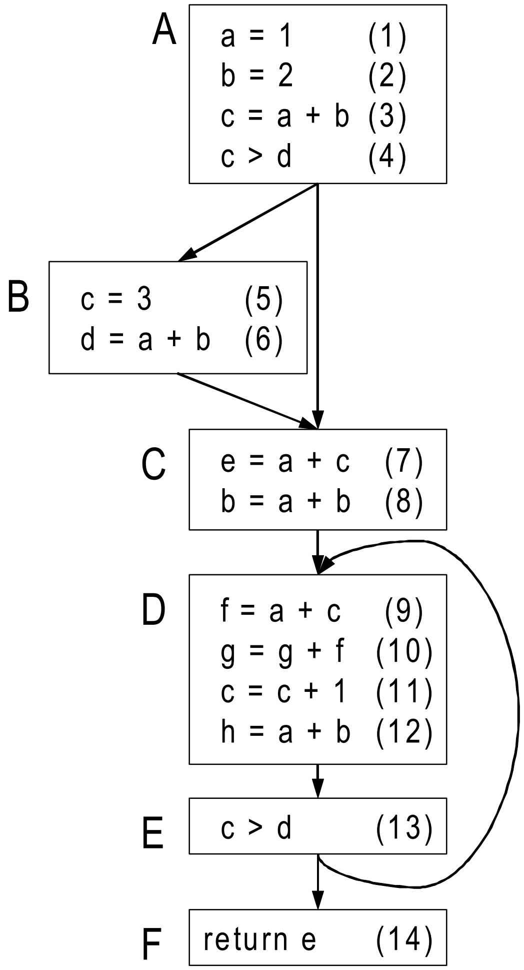


Figure 1: (this diagram is reprinted on page 11, which can be detached)

Question 3. Basic Blocks 4 points

In the above control flow graph (CFG) of basic blocks (Figure 1)

a) All possible statements have been combined to form the right number of basic blocks

b) The blocks A and C can be combined together to form a basic block

c) The blocks B and C can be combined together to form a basic block

d) The blocks D and E can be combined together to form a basic block

e) The blocks E and F can be combined together to form a basic block

Question 4. Reaching Definitions 8 points

What are the reaching definitions at the beginning of the basic block D of Figure 1?
(mark the reaching definition numbers by crossing off the corresponding boxes with the statement numbers)

(1)	(2)	(3)	(5)
(6)	(7)	(8)	(9)
(10)	(11)	(12)	

Question 5. Available Expressions 8 points

What expressions are available at the beginning of the basic block D of Figure 1?
(mark available expressions by crossing off the corresponding box)

a+b	c > d	a+c
g+f	c+1	

Question 6. Dead Code Elimination 8 points

If you ran Live Variable Analysis followed by Dead Code Elimination on the CFG of Figure 1, what statements will be eliminated? Assume no variable is live at exit.
(write down the statement number of statements (min 0, max 4) that can get eliminated)

□ □ □ □

Question 7. Constant Propagation 11 points

If you ran Reaching Definitions Analysis followed by Constant Propagation on the CFG of Figure 1 what constants can be replaced?
Note: this question uses the un-altered CFG in Figure 1
(write down the constant value on the appropriate boxes and leave the rest as blank)

a in (7) with □

c in (7) with □

a in (8) with □

b in (8) with □

a in (9) with □

c in (9) with □

g in (10) with □

f in (10) with □

c in (11) with □

a in (12) with □

b in (12) with □

Question 8. More Constant Propagation 6 points

After the first Constant Propagation iteration of Question 7, you ran Algebraic Simplification, Copy Propagation, Dead Code Elimination until convergence. Can you identify one additional replacement that was not found during the first Constant Propagation iteration?
(write down the variable name, statement number and the constant value)

□ in □ with □

Question 9. Common Subexpression Elimination 8 points

If you ran Available Expression Analysis followed by Common Subexpression Elimination algorithm on the CFG of Figure 1, what expressions can be eliminated?
Note: this question uses the un-altered CFG in Figure 1
(Write down the statement number of the source common sub expression that is used replace the statement in the given line. Leave the space blank if it cannot be eliminated by CSE.)

a+b in (3) with □

a+b in (6) with □

a+c in (7) with □

a+b in (8) with □

a+c in (9) with □

a+b in (12) with □

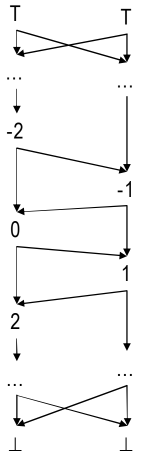
Question 10. Lattice Patterns 5 points

Which of the following are considered a complete lattice?
(Circle all that apply)

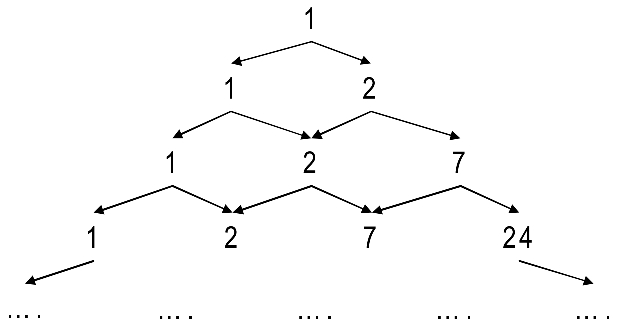
a) All positive integers

b) The set of all real numbers and positive and negative infinity.

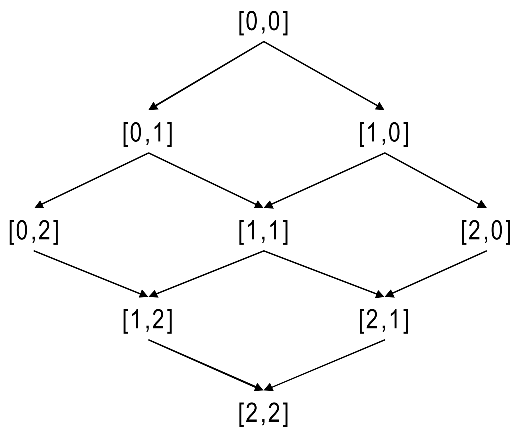
c) □



d)



e)



We have added a new data type into Decaf called BIT. The type descriptors of the BIT type also indicate the number of bits, which is between 1 and 16. For example:

```
BIT(8) A; □
BIT(16) B; □
BIT(6) C; □
BIT(2) D; □
```

The only operations allowed on the BIT data types are bitwise AND, OR and assignment.

```
A &= B; □
 □
B |= C; □
 □
D = 0x1011; □
```

Note that these statements take values with different number of bits. The syntax for these operations is:

- if the number bits in the right hand side variable is the same as the left and side, then do the normal operation
- if the number of bits in the left hand side is higher, then zero extend the right hand side value to have the same number of bits
- if the number of bits in the left hand side is lower, then truncate the right hand side value so that it has the same number of bits

In `A &= B`, where `A` is 8 bits and `B` is 16 bits, the value from `B` is truncated to 8 bits before used in the statement.

We have observed that many variables have higher number of bits than is required by all its uses. Thus most uses truncate the value of the variable before using it. It seems inefficient to store all these extra bits. We have proposed a new Bit Length Analysis where we calculate the number of bits that are live for each variable at each program point. We plan to run this analysis and then use the fixed-point solution to reduce the size of declarations of type BIT.

The Bit Length Analysis is a backwards dataflow analysis.

Question 11. The Lattice 10 points

We are only concerned with a single variable in our analysis. In this data-flow problem, at each program point, a *variable* is represented by a value from a lattice. Draw the lattice used for the dataflow problem for a single variable.

Question 12. Initial Condition 6 points

We are only concerned with a single variable A. If we are using the worklist algorithm to arrive at a fixed-point, what is the initial value of the IN[*exit*], where *exit* is the single exit node. Assume that A is not live after *exit*.

IN[exit] =

Question 13. Transfer Functions 6 points

A transfer function describes what effect a statement (or a sequence of statements of a basic block) has on the value of the dataflow analysis. For a backwards analysis, a transfer function defines the IN value in terms of the OUT value.

Our analysis is formulated such that it only tracks the bit length of variable B.

For the statement S,

```
S: A &= B
```

What is the correct transfer function? Assume that A's declared bit length is L_A and B's declared bit length is L_B.
(Circle the correct answer)

a) $F_S(\mathrm{OUT}[S]) = \mathbf{MIN}(\mathrm{OUT}[S], L_A)$

b) $F_S(\mathrm{OUT}[S]) = \mathbf{MIN}(L_B, \mathbf{MAX}(\mathrm{OUT}[S], L_A))$

c) $F_S(\mathrm{OUT}[S]) = \mathbf{MAX}(\mathrm{OUT}[S], L_A) - L_B$

d) $F_S(\mathrm{OUT}[S]) = \mathbf{MIN}(L_A, \mathbf{MIN}(\mathrm{OUT}[S], L_B))$

e) $F_S(\mathrm{OUT}[S]) = L_B - L_A + \mathrm{OUT}[S]$

Question 14. Confluence (Join) Operator 6 points

Again, assume that we formulate our analysis for a single variable. Given a block with multiple outgoing edges, what is the confluence (join) operator?
(Circle the correct answer)

a) SUM □

b) MIN □

c) MAX □

d) AVERAGE □

e) 16 □

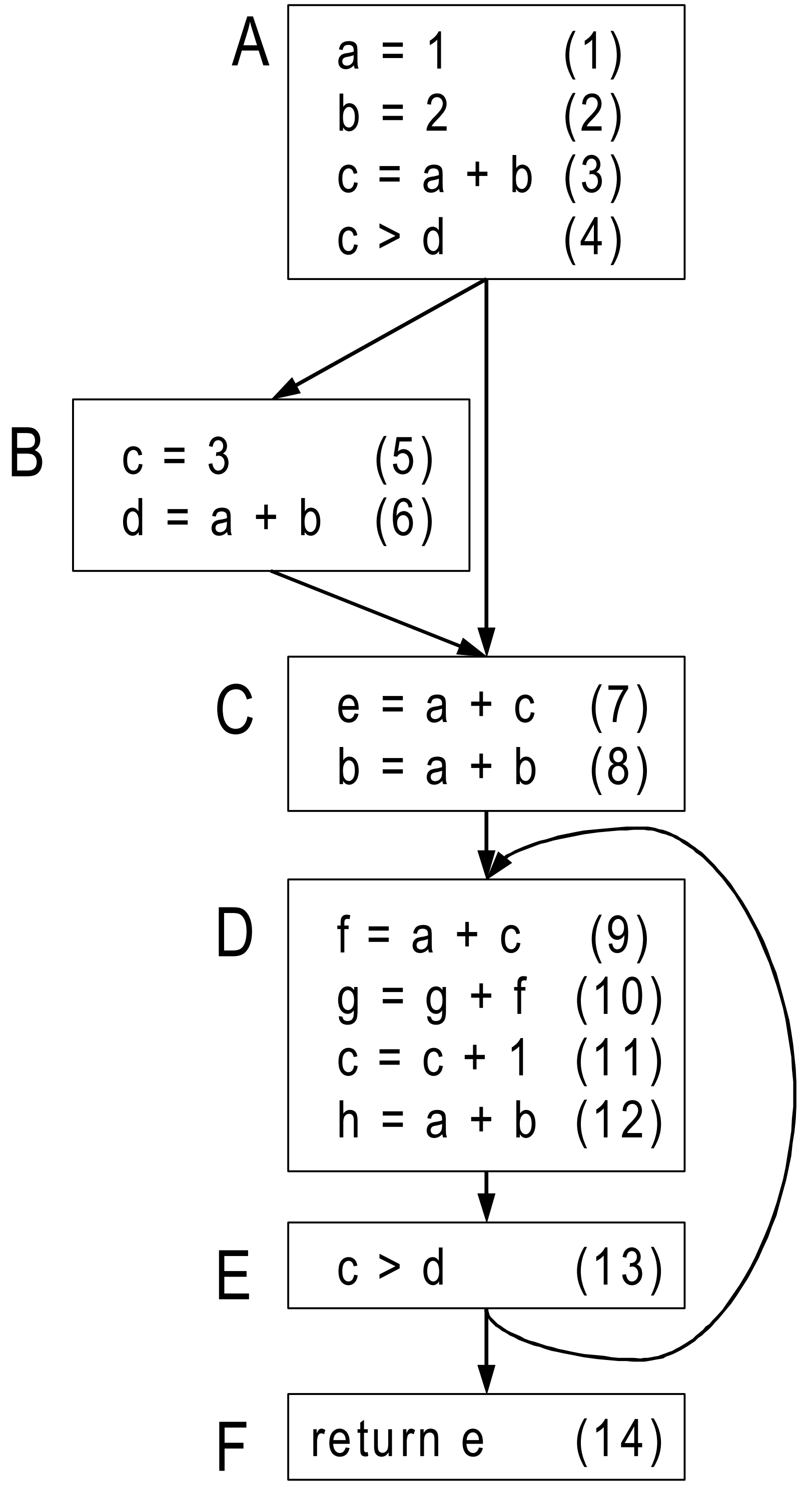

A
a = 1 (1)
b = 2 (2)
c = a + b (3)
c > d (4)
B
c = 3 (5)
d = a + b (6)
C
e = a + c (7)
b = a + b (8)
D
f = a + c (9)
g = g + f (10)
c = c + 1 (11)
h = a + b (12)
E
c > d (13)
F
return e (14)

Figure 1

MIT OpenCourseWare
http://ocw.mit.edu

6.035 Computer Language Engineering
Spring 2010

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.