CSE 531S Translation of Computer Languages **Fall 2011**

Midterm Exam

Given: 24 October 2011 Due: 1 November 2011

Although this exam is take-home, you are not permitted to discuss any aspect of this exam with anybody except the instructor. You may use any reference materials or computer programs you wish in solving this exam, but the solutions you submit must be your own work.

Partial credit will be given where sufficient detail is provided.

By turning in this exam, you agree that you have abided by the Statement of Student Academic Integrity.

Name:		
Student ID:		
Problem Number	Possible Points	Received Points
1	10	
2	15	
3	15	
4	15	
5	45	
Total		

1. (10 points) Prove or disprove each of the theorems below about meet lattices, assuming only the following axioms:

   Axioms:

$$\begin{array}{lcl} x \preceq y & \iff & x \wedge y = x \\ x \wedge y & \preceq & x \\ x \wedge y & \preceq & y \end{array}$$

   (a) (5 points)

$$a \preceq b \text{ and } b \preceq c \Longrightarrow a \wedge c = a$$

   (b) (5 points)

$$a \preceq b \text{ and } c \preceq d \Longrightarrow a \wedge c \preceq b \wedge d$$

2. (15 points)

(a) (5 points)

i. (3 points) Under what conditions is a data flow framework *monotonic*?

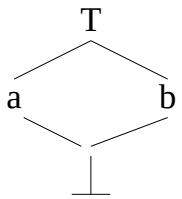
ii. (2 points) What are the advantages of a monotonic framework?

(b) (10 points) For each of the data flow frameworks described below, assess and state whether the framework is *monotonic* or not.

- If you believe the framework is monotonic, show your work by a complete proof of its monotonicity. The frameworks are all finite so you can do this exhaustively.
- If you believe the framework is not monotonic, support that belief by
  - Pointing out the transfer functions that behave non-monotonically.
  - Constructing a flow graph (of at most two nodes) for which data flow analysis would not terminate.

i. (5 points)

Lattice:

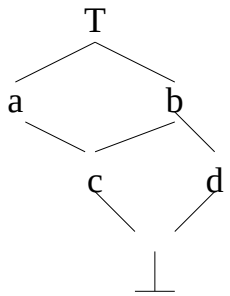


Transfer Functions:

Function	Input x			
	$\top$	a	b	$\bot$
$f(x) =$	$\top$	b	a	$\bot$

ii. (5 points)

Lattice:

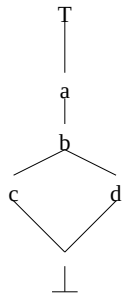


Transfer Functions:

	Input x					
Function	$\top$	a	b	c	d	$\bot$
$f_1(x) =$	a	a	$\top$	b	c	$\bot$
$f_2(x) =$	$\top$	$\top$	$\top$	$\bot$	$\bot$	$\bot$

3. (15 points) Consider the following data flow framework:

Lattice:



Transfer Functions:

	Input x					
Function	$\top$	a	b	c	d	$\bot$
$f(x) =$	$\top$	a	b			$\bot$

(a) (3 points) Generally, under what conditions is a data flow framework distributive?

(b) (2 points) What are the advantages of a distributive framework?

(c) (10 points) Complete the transfer function table above so that the framework is **not** distributive. Your solution must be monotonic, and you cannot use $\top$ or $\bot$ to fill in the table. Finally, show why the resulting framework is not distributive.

4. (15 points) Consider the following data flow framework:

Lattice:

Transfer Functions: To be specified

(a) (3 points) Generally, under what conditions is a data flow framework rapid?

(b) (2 points) What are the advantages of a rapid framework?

(c) (5 points) Complete the transfer function table below, so that the data flow framework with the lattice shown above **is** rapid. Your solution must be monotonic, and you cannot use $\top$ or $\bot$ to fill in the table.

	Input x					
Function	$\top$	a	b	c	d	$\bot$
$f(x) =$	$\top$					$\bot$

(d) (5 points) Complete the transfer function table below, so that the data flow framework with the lattice shown above **is not** rapid. Your solution must be monotonic, and you cannot use $\top$ or $\bot$ to fill in the table.

	Input x					
Function	$\top$	a	b	c	d	$\bot$
$f(x) =$	$\top$					$\bot$

5. (45 points) A variable is *useful* if

   - Its current value can be output using a `print` statement. For example, just before the node containing:

     ```
     print x
     ```

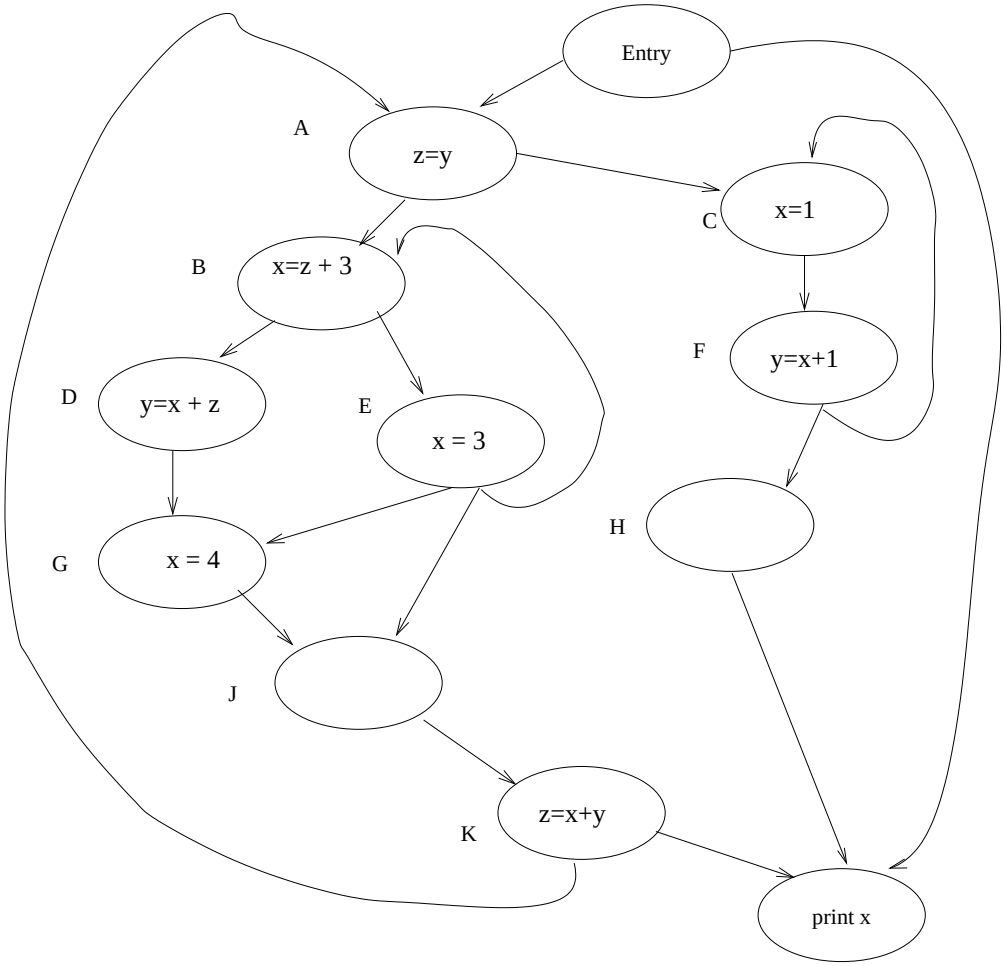
     the variable `x` is useful.

   - Its current value participates in a computation of a useful variable. For example, consider a node containing the following:

     ```
     x = a + b
     ```

     If `x` is a useful variable just after that node, then `a` and `b` are useful variables just before that node, and `x` is not useful just before that node (its current value cannot be referenced given the assignment statement in the node).

   In this problem, you eventually consider solving the USEFUL VARIABLES problem for the following flow graph:



   But before you do that, there are some seemingly irrelevant questions you must answer ....

(a) (1 points) Depth-first number the flow graph on the graph itself.

(b) (4 points) What is the corresponding topological node order for the graph? Respond by specifying vertex labels, not depth-first numbers (but use your depth-first numbering to answer the question).

(c) (5 points) Draw the dominator tree for this flow graph. (Again, respond with vertex labels).

(d) (5 points) Fill in the following table to show the dominance frontier for each node in the flow graph.

X	DF(X)
A	
B	
C	
D	
E	
F	
G	
H	
J	
K	

Continued on next page...

(e) Based on the kinds of statements shown in the flow graph above, formulate a data flow framework for this problem by answering the following questions:

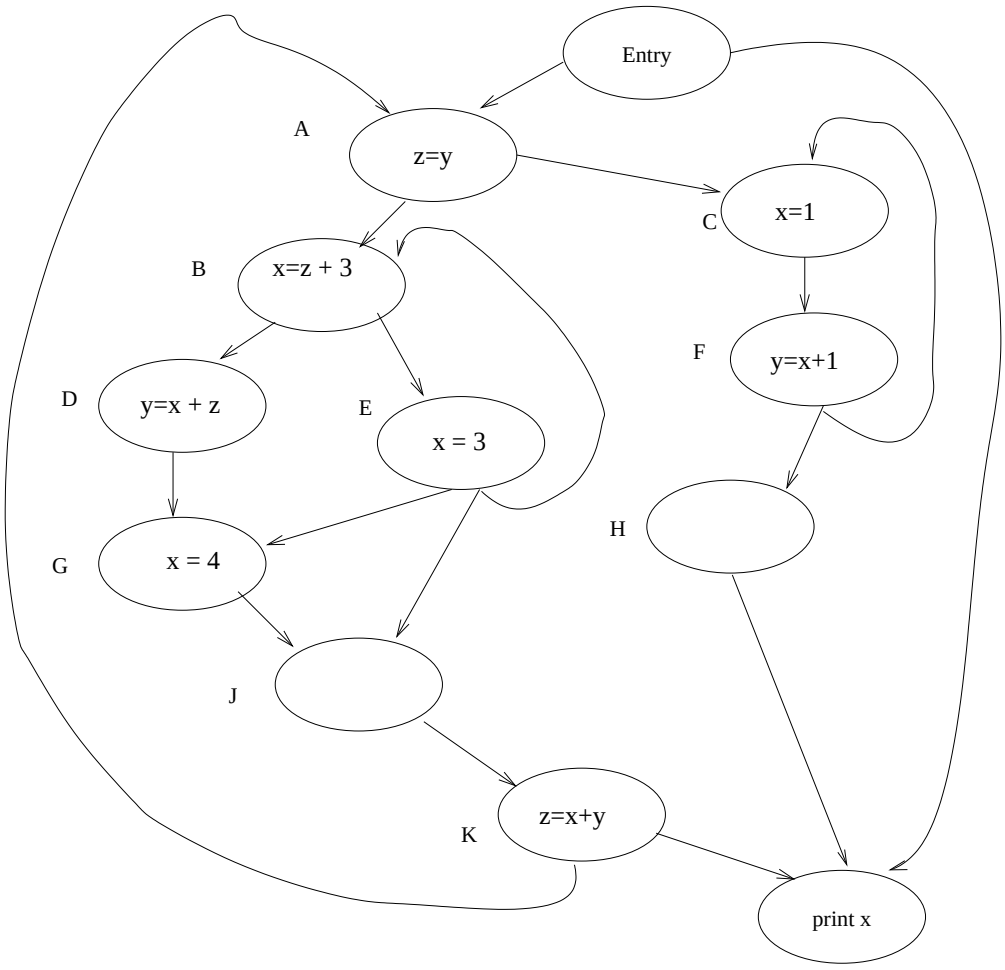
i. (1 points) Is your framework forward or backward?

ii. (4 points) The domain for this problem is the the set of all subsets of variables used in the program. What is the lattice for this problem? More specifically, define $\top$, $\bot$, the $\wedge$ operator, and the $\preceq$ operator:

iii. (5 points) Describe the data flow transfer functions for nodes of the type shown in the graph above. Be careful! Do not limit your treatment to exactly the nodes' contents in the graph, but base your treatment on the *kinds* of statements (assignments, prints, and nothing).

(f) (10 points) Fill in the table below for the solution to the useful variables problem. Your solution should reflect which among the variables `x, y, z` are useful *on entry* to the specified node, before any statements in the node have executed. Show your work on the copy of the flow graph below the table.

Node X	Useful Variables at top of Node X
A	
B	
C	
D	
E	
F	
G	
H	
J	
K	



(g) (5 points) Prove or disprove that this framework is rapid.

(h) (5 points) Prove or disprove that this framework is distributive.