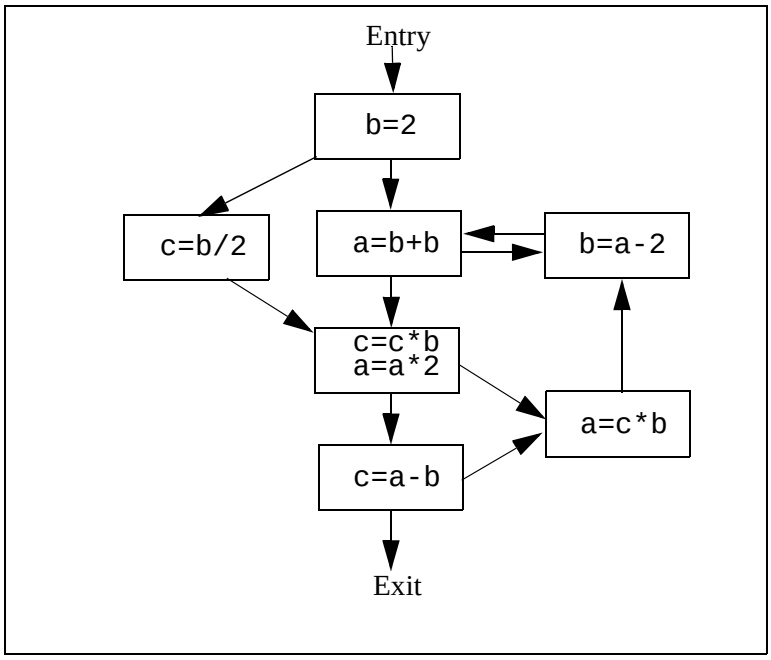
CS 243 Assignment 3

Data Flow Foundation and Analyses

Due: February 6, 11:00 am

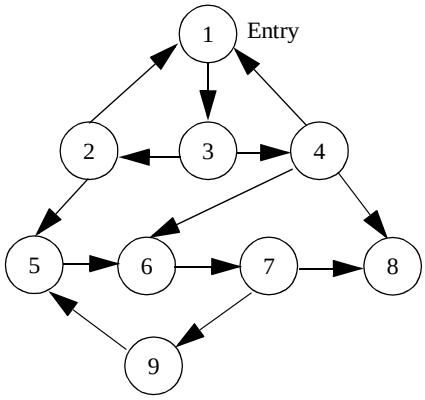
This is a written assignment, every student must hand in his or her homework. Bring your homework to class on Wednesday, February 6, or give your homework to the courier.

1. Apply the constant propagation algorithm described in class to the following program. Replace all the references to variables and operations with constants whenever possible.



2. This question is about the data flow algorithm that we defined in class for finding dominators.
   a. Prove that the data flow framework formulated is monotone.
   b. Is the data flow framework distributive? Why?
   c. In general, what is the significance between a distributive vs. a monotone framework? (This is not specific to the dominator data flow problem).

3. Consider the following flow graph:



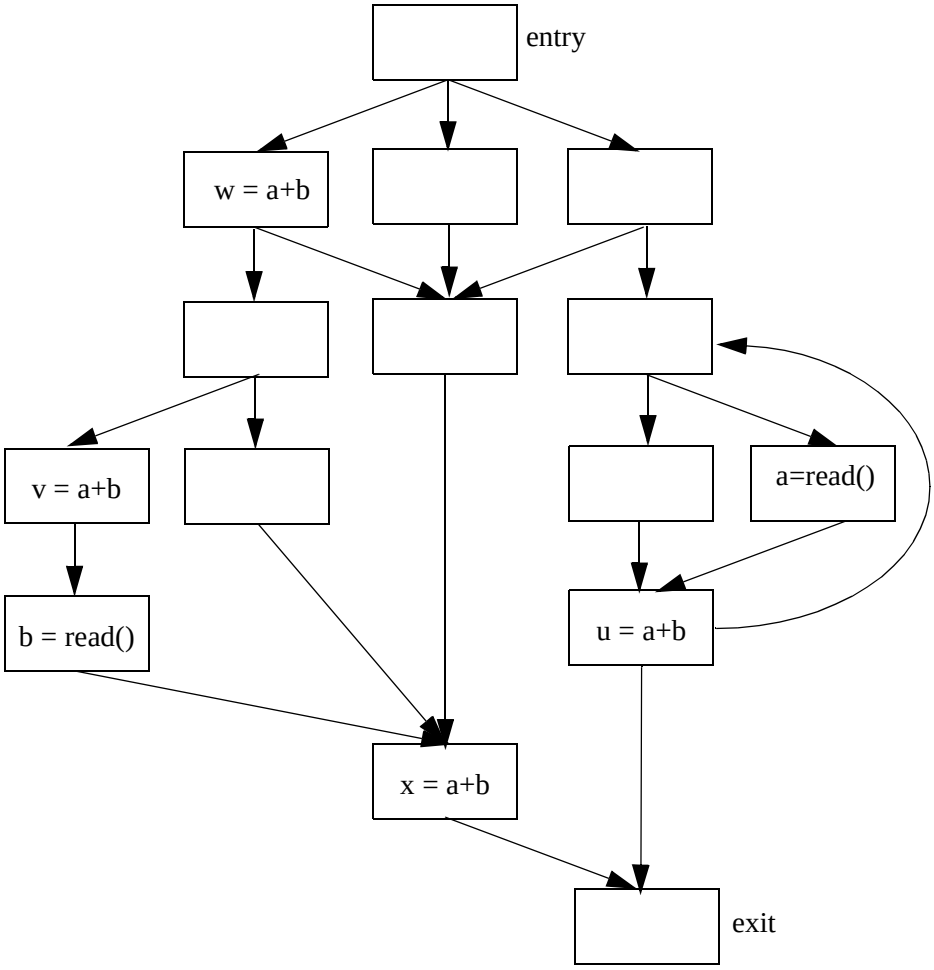
   a. Show the dominator tree for this flow graph.

   b. What are the back edges in this flow graph?

   c. Find the natural loop associated with each of the back edges.

   d. Is this program reducible?

4. Apply lazy code motion to the following program. You do not need to show the intermediate steps, just show the optimized code. You may add basic blocks to the flow graph, but only show those that are not empty in your solution.



5. Critical sections are a very fundamental concept in concurrent programming. A communicating process must issue a LOCK operation before entering a critical section and issue an UNLOCK operation on exit in order to guarantee mutual exclusion.

   For simplicity, assume that there is only one lock in the system, and therefore the LOCK and UNLOCK operations do not have any arguments. You can ignore all the other operations in the program. In a correct execution sequence, every LOCK operation must be immediately followed by an UNLOCK operation, and every UNLOCK operation must be preceded by a LOCK operation.

   Your task is to develop a static program analysis that will issue the following four types of warnings:

   - Warning I: LOCK $\rightarrow$ LOCK. Issue a warning on a LOCK operation if it can potentially follow another LOCK operation.
   - Warning II: UNLOCK $\rightarrow$ UNLOCK. Issue a warning on an UNLOCK operation if it can potentially follow another UNLOCK operation.
   - Warning III: LOCK $\rightarrow$ EXIT Issue a warning on a LOCK operation if the program may terminate without performing an UNLOCK.
   - Warning IV: ENTRY $\rightarrow$ UNLOCK. Issue a warning on an UNLOCK operation if it may be executed before any LOCK operations.

   You may assume that there are no procedure calls in the input programs. Describe your analysis. You may want to define one or more data flow algorithms to solve this problem. If you solution uses data flow analysis, specify the algorithm fully by answering the following questions:

   i. What is the direction of your data flow analysis?

   ii. What is the set of values in the semi-lattice? **Explain what the values denote.**

   iii. Draw a diagram of the lattice, identifying the top and bottom elements clearly.

   iv. How would you initialize the iterative algorithm?

   v. Define the transfer function of a basic block.

   vi. How would you initialize the information at the entry/exit nodes?

   vii. Is your data flow framework monotone? (No explanation is necessary).

   viii. Is your data flow framework distributive? (No explanation is necessary).

   ix. Will your algorithm necessarily converge? If so, why?

   Make sure you specify how you identify the operations that have raised the warning.