ECE 468 & 573

Problem Set 3: Common sub-expression elimination, local register allocation.

For the following problems, consider the following piece of three-address code:

```
1.  READ(A)
2.  READ(B)
3.  C = A + B
4.  D = A + B
5.  E = C * D
6.  T1 = A + C
7.  T2 = T1 + E
8.  A = T2 + B
9.  F = A + B
10. G = C * D
11. T3 = F + G
12. WRITE(T3)
```

1. Show the result of performing Common Subexpression Elimination (CSE) on the above code. Rather than generating assembly, just give new 3AC. If you're reusing the results of an expression, just generate code that looks like X = Y, where Y is the variable/temporary that held the result of the original calculation.

2. Suppose A and C were aliased. How would that change the results of CSE?

3. For each instruction, show which variables are live *immediately after the instruction.* (Use the original code, not the CSE code)

4. Assume that E is a global variable, and the given code is the code from a single function in a program with many functions. What changes in your liveness analysis?

5. (Go back to assuming the above code is the entire program) Perform bottom-up register allocation on the code for a machine with four registers. Show what code would be generated for each 3AC instruction. Use LOAD X Rx to load from a variable/temporary into a register, STORE Rx X to store from a register into a variable/temporary, Rx = Ry + Rz for addition, and Rx = Ry * Rz for multiplication.

   When choosing registers to allocate, always allocate the lowest-numbered register available. When choosing registers to spill, choose the non-dirty register that will be used farthest in the future. In case all registers are dirty, choose the register that will be used farthest in the future. In case of a tie, choose the lowest-numbered register.

6. Repeat the process for *3* registers.