

**ECE540S: Optimizing Compilers**  
**Miterm Exam, March 2, 2004**

**Closed Book, Closed Notes, No Electronic Devices**  
**60 Minutes**

**NAME** \_\_\_\_\_

**STUDENT NUMBER** \_\_\_\_\_

**EMAIL ADDRESS** \_\_\_\_\_

**Q1: / 15**

**Q2: / 10**

**Q3: / 15**

**Q4: / 10**

**Q5: / 15**

**Q6: / 10**

**Q7: / 15**

**Q8: / 10**

**TOTAL: / 100**

## CONTROL FLOW ANALYSIS (25 marks)

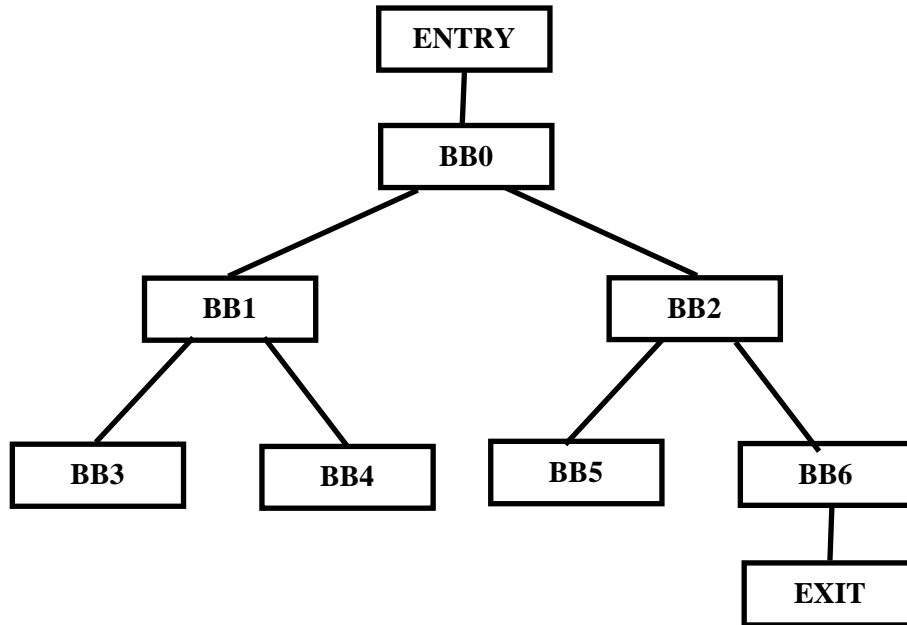
1) Use the following code to answer parts (a) and (b).

```
S1:  X = X + 10
S2:  IF (X > 10) GOTO S9
S3:  GOTO S8
S4:  IF (Y > 5) GOTO S9
S5:  Y = Y + 1
S6:  X = X + 1
S7:  IF (X < 100) GOTO S4
S8:  IF (Y < 100) GOTO S2
S9:  PRINT X, Y
S10: RETURN
```

a) Identify the leader instructions and their corresponding basic blocks. You may do this directly on the code snippet. Draw the control flow graph below (10 marks):

b) Identify the back-edge(s) in the control flow graph drawn in part (a). Provide them below using the form  $T \rightarrow H$ , where  $T$  is the basic block at the tail of the edge and  $H$  is the basic block at the head of the edge (5 marks).

2) You are given a routine P with basic blocks BB0 ... BB6. Use the following dominator tree below to determine if the statements made in parts (a) – (e) must be true or may be false.



a) There is no unreachable code in the routine (true/false) (2 marks).

b) There is no path from BB4 to BB2 (true/false) (2 marks).

c) There is a path from BB1 to BB6 (true/false) (2 marks).

d) BB1 to BB2 cannot be in the same natural loop (true/false) (2 marks).

e) BB4 cannot be the header of a natural loop (true/false) (2 marks).

## DATA FLOW ANALYSIS (25 marks)

3) Assume that there are 2 operations  $\text{lock}(v)$  and  $\text{unlock}(v)$ , where  $v$  is a lock variable. It is legal to unlock a variable  $v$  that has not been lock-ed, but it is never legal to lock the same variable  $v$  twice without at least 1 intervening unlock of  $v$ . Trying to lock a variable that you have already lock-ed would lead to a deadlock.

To ensure that user code is safely written, you are asked to design an iterative data flow solver to address this problem. You decide to implement a solver that will determine, for every point  $p$ , what lock operations reach that point. Using this information you'll then be able to know if each lock is safe, i.e. it is not reached by another lock of the same variable. A  $\text{lock}$  of a variable  $v$  reaches a point  $p$  if there is no intervening  $\text{unlock}(v)$  between the point of the  $\text{lock}(v)$  and  $p$ .

Answer parts (a) – (e) based on your design for such an iterative data flow solver.

a) What would your bit vectors represent (2 marks)?

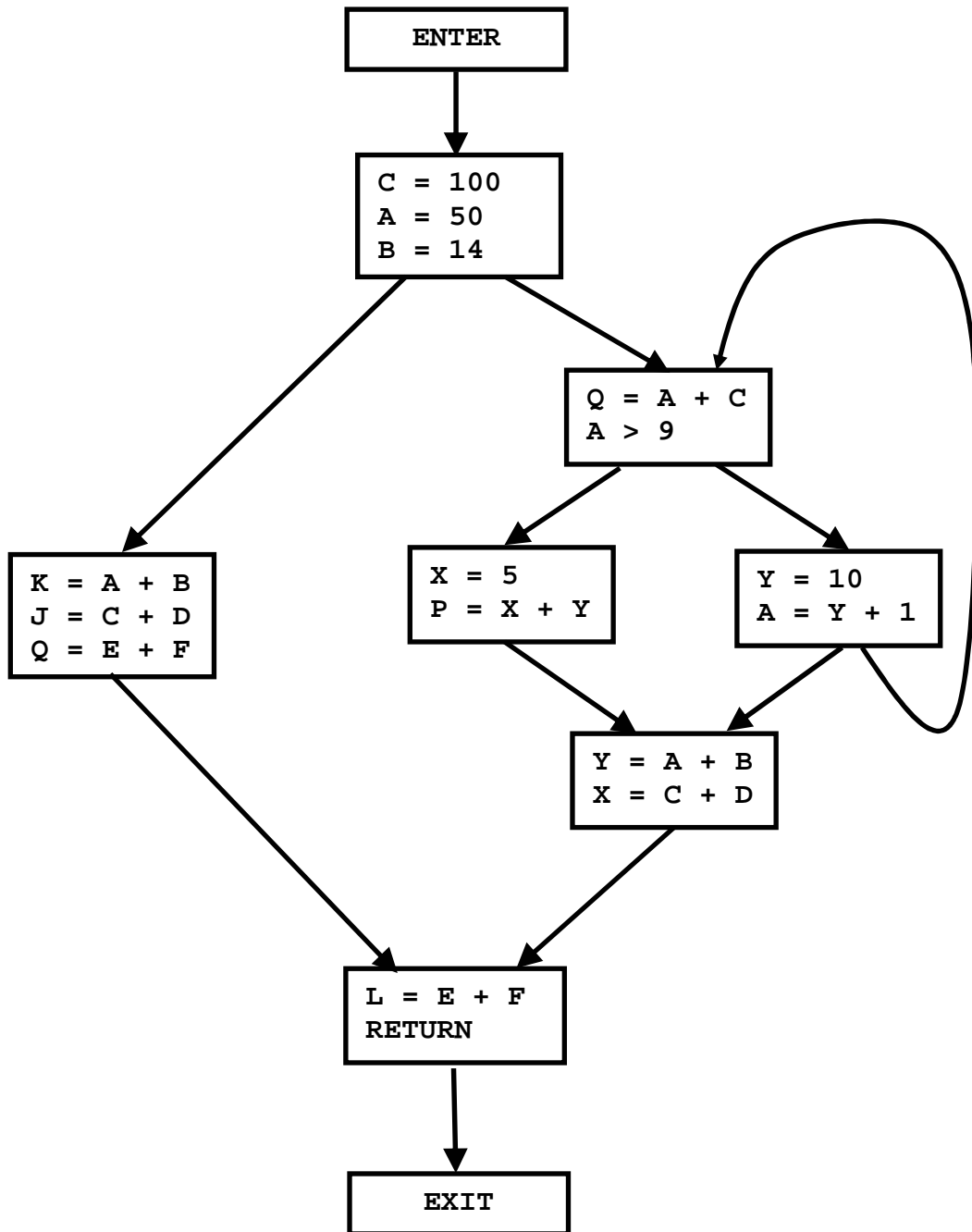
b) Would you implement your solver as a forward or backward problem (2 marks)?

c) What is your meet operator (2 marks)?

**d) Define the Gen(b) set and Kill(b) set for a basic block b (4 marks).**

**e) Provide the data flow equations for calculating the In and Out sets for a basic block b (5 marks):**

4) Perform Very Busy Expressions analysis on the following CFG. Label each basic block with the final contents of its In and Out sets (10 marks).



**REDUNDANCY ELIMINATION (35 marks)**

5) Use the following code to answer parts (a) and (b).

```
X = Y + Z
K = Y + Z
G = K + A
H = X + A
J = G + H
H = K + A
G = G + H
H = G + H
```

a) Show the above code after removing redundant code by using local value numbering. Perform no other optimizations (7.5 marks):

b) Show the above code after removing redundant code by using local common sub-expression elimination (begin with the original code, not the result from part a). Perform no other optimizations (7.5 marks):

6) Use the following code to answer parts (a) and (b). Assume that the code below is only entered through the instruction at L1.

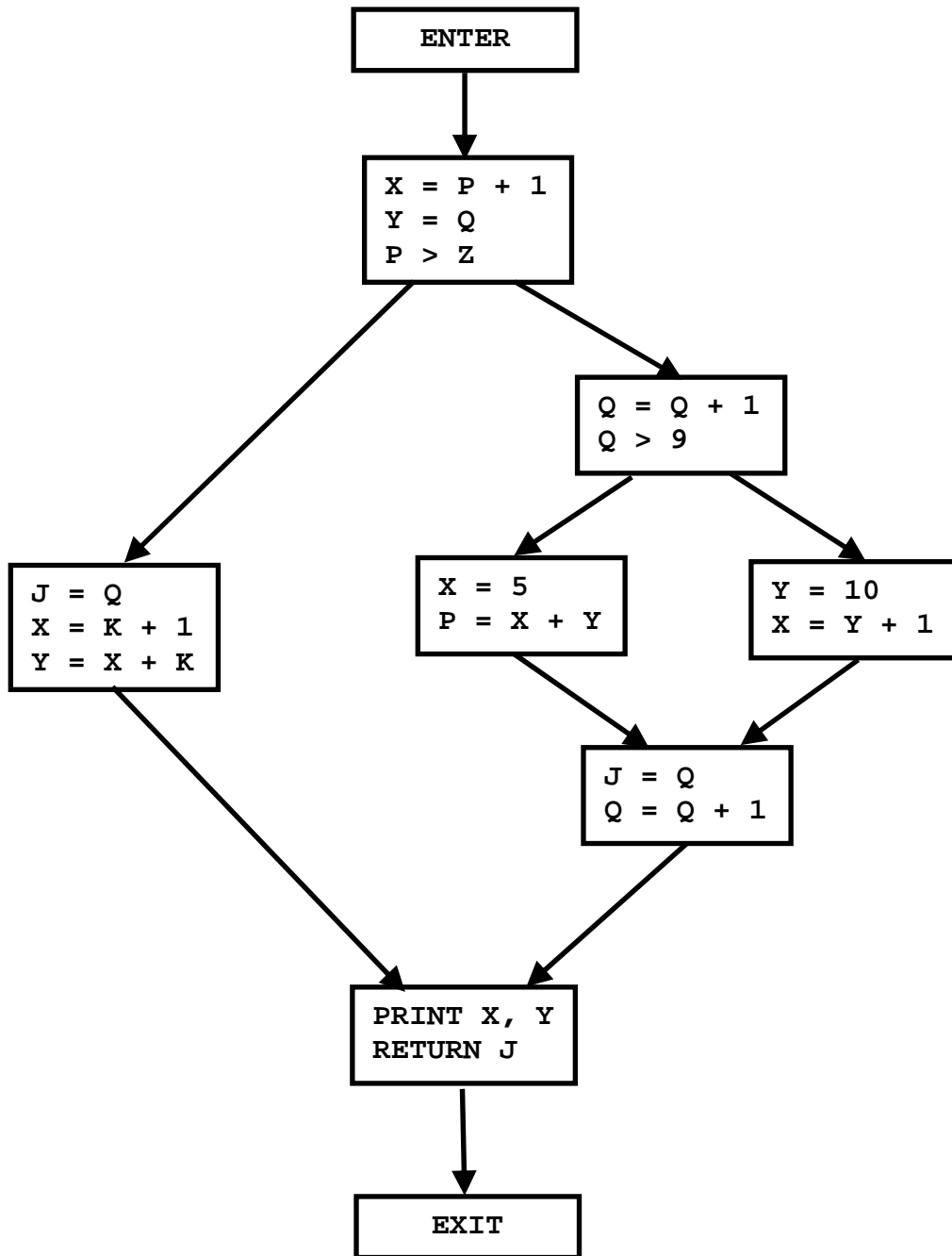
```
L1:  K = 10
      X = M + 1
      L = L + 1
      Q = J + 1
      IF (K > L) GOTO L3
      M = K + 1
L3:  J = K + 1
      IF (X > 100) GOTO L1
```

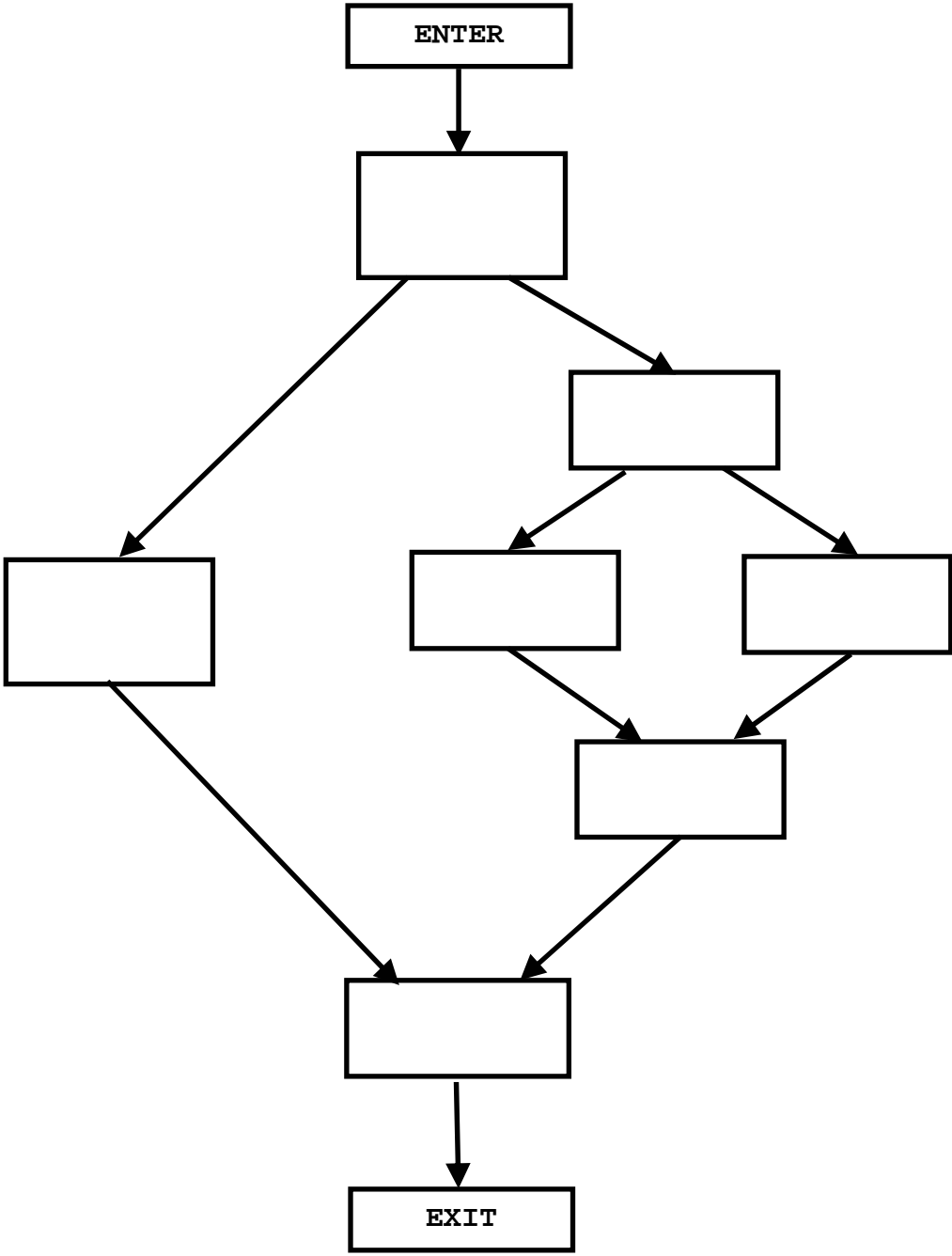
a) Identify the instructions in the above code that always compute the same value in each iteration of the loop (i.e. the computations that are loop invariant) (5 marks).

b) Re-write the code below after performing loop invariant code motion (5 marks):

DEADCODE ELIMINATION (15 marks)

7) Re-write the following code with all dead-code eliminated (there is an empty CFG available on the next page for this purpose). If all statements are removed from a basic block, just leave the block empty. Perform no other optimizations (15 marks).





**MISCELLANEOUS (10 marks)**

**8) Provide short answers to the following questions (2 marks each):**

**a) If you are given an irreducible control-flow graph, how can you find the back-edges?**

**b) Should you perform copy propagation before or after common sub-expression elimination?**

**c) What is the purpose of an intermediate representation?**

**d) In what way is reaching definitions analysis conservative?**

**e) Why might it be more difficult to perform constant folding on floating point values than on integer values?**