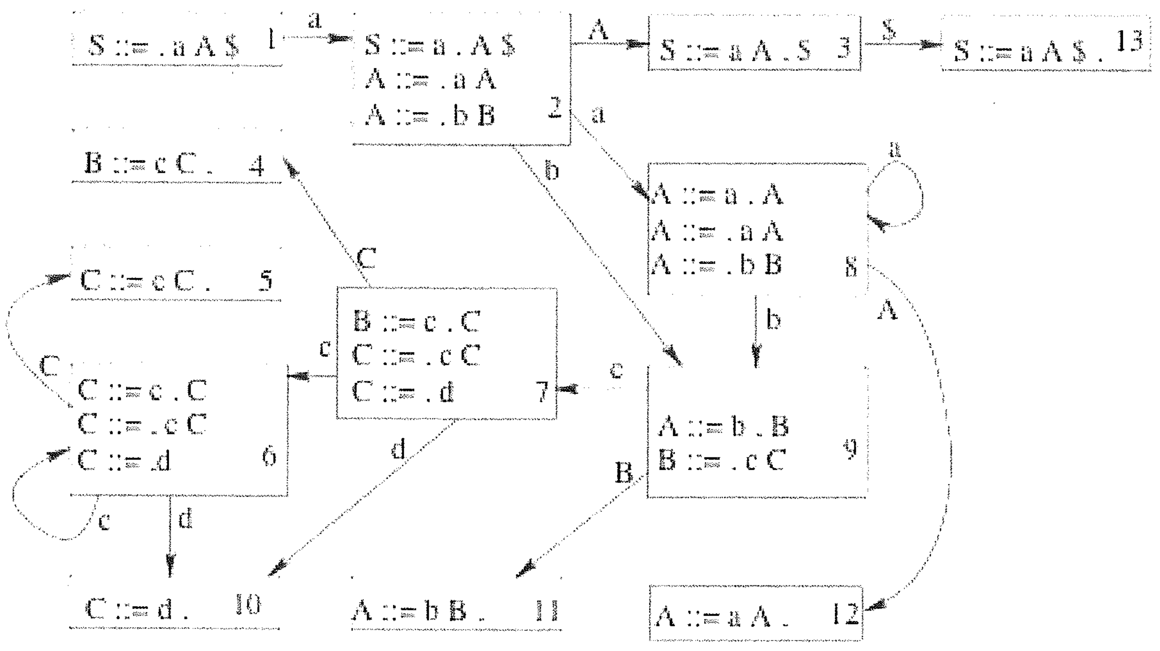
CE-2
August 2010 QE

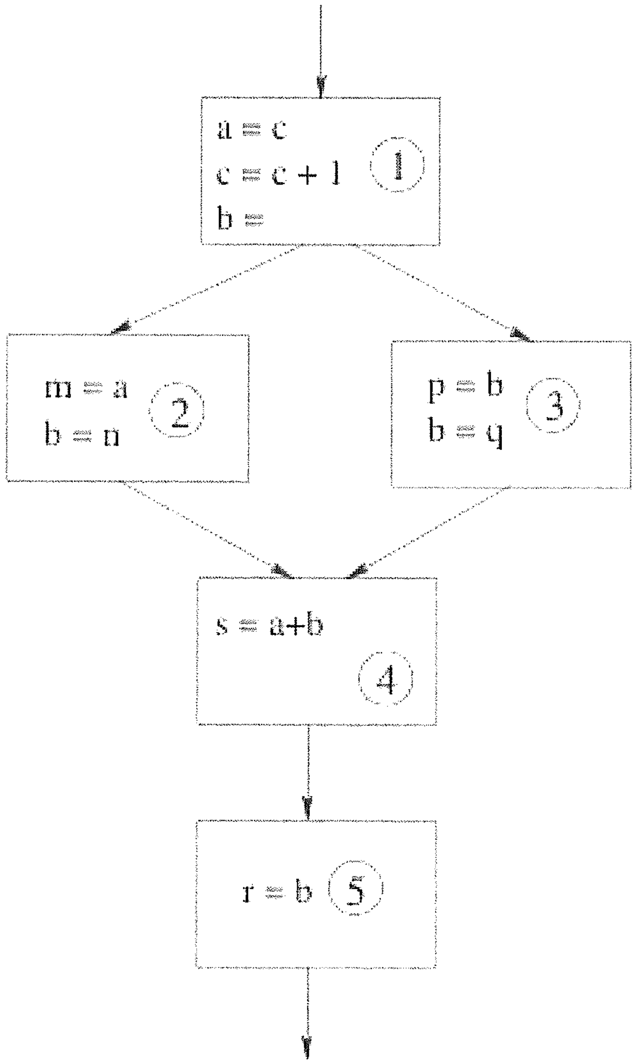
1. **(6 points)** Consider the language L_1 defined by the regular expression 0^n1^n, where n is some *fixed* integer. Express, in terms of n, the minimal number of states contained by a deterministic finite state machine that can recognize this regular expression.
2. **(6 points)** Let L_2 be the languages described by 0^n1^n, $n>0$.
   a. Give a context free grammar for L_2. Say if L_2 can be recognized by a deterministic finite state machine
   b. Briefly (30 words or less) state why or why not.
3. Use the CFSM at the bottom of the page to answer parts **a**, **b** and **c** of this question.
   a. **(6 points)** What are the products with non-terminal A on the left hand side (i.e. productions of the form "*A ::= ...*" in the grammar for which the CFSM was constructed?
   b. **(6 points)** Let the symbol stack be *a a b c* (where *c* is the top of the stack), the state stack is *1 2 8 9 7*, and the next symbol is *d*. What is the next parser action, and the state of the symbol and state stack after that parser action? When specifying the parser action, you only need to say the action; you do not need to specify the production involved, etc.
   c. **(6 points)** Let the symbol stack be *a a b B* (where *B* is the top of the stack), the state stack is *1 2 8 9 11*, and the next symbol is *$*. What is the next parser action, and the state of the symbol and state stack after that parser action? When specifying the parser action, you only need to say the action; you do not need to specify the production involved, etc.



4. **(25 points)** Using the flow graph below, perform a *live variable analysis*. A variable is *live* in some block *B* if it is used along some path out of *B* before being written along the path. To answer this question, fill in a table like the one below ***in your blue books – do not fill in the table below.***

Block	GEN	KILL	IN	OUT
1				
2				
3				
4				
5				

Write in Exam Book Only



5. Given the loop nest

```
for (i=0; i < n; i++) {
  for (j=0; j < n; j++) {
     a[2*i+1, j] = b[i+1,j]
     b[i, j] = a[2*i-1,j]
  }
}
```

   a. **(8 points)** Describe the dependences on the "**a**" array in the loop nest above. Either say "no dependence", or, if a dependence exists, give the type (flow or true, output or anti), the direction and the distance.
   b. **(8 points)** Describe the dependences on the "**b**" array in the loop nest above. Either say "no dependence", or, if a dependence exists, give the type (flow or true, output or anti), the direction and the distance.
   c. **(8 points)** Which loop(s), if any, can be parallelized in the loop nest above, without performing any other transformation?
   d. **(5 points)** If one or both of the loops in the nest cannot be parallelized is there a transformation that will either increase the number of loops that are parallel, or will enhance (i.e. make more efficient) the exploitation of existing parallelism? If so, name a transformation that does this, show the resulting code, and briefly explain (in 30 words or less) how the parallelism is increased or enhanced. If no such transformation exists, simply write "none" as your answer.

6. The Cray-1 computer was the fastest computer of its time. It did not have caches. For each transformation listed below, tell whether it is more useful, less useful, or the same usefulness on a Cray-1 as on a machine with a cache. Explain why in 30 words or less.
   a. **(4 points)** Register allocation
   b. **(4 points)** Loop interchange
   c. **(4 points)** Instruction scheduling
   d. **(4 points)** Loop tiling

Write in Exam Book Only