Computer Programming Lab **14/9/2007**

Writing Recursive Functions without Recursions

Normally a non-recursive version of a program will execute more efficiently compared to a recursive implementation in terms of space and time. This is because the overhead involved in entering and exiting a block is avoided in the non-recursive version. To convert a recursive program into a non-recursive program it is advised to first develop a solution using stacks from the recursive definitions. Then the programmer can optimize the code to reduce needless stacking activities, which a compiler may be unable to perform.

a) Write a recursive C-code to solve the Tower of Hanoi problem (discussed in the class). Implement using a function:

tower(int n, char frompeg, char topeg, char auxpeg).

Here "n" means the number of disks, "frompeg" indicates the starting peg, say A. "topeg" indicates the end peg, say "B", while the auxiliary peg is denoted by "C".

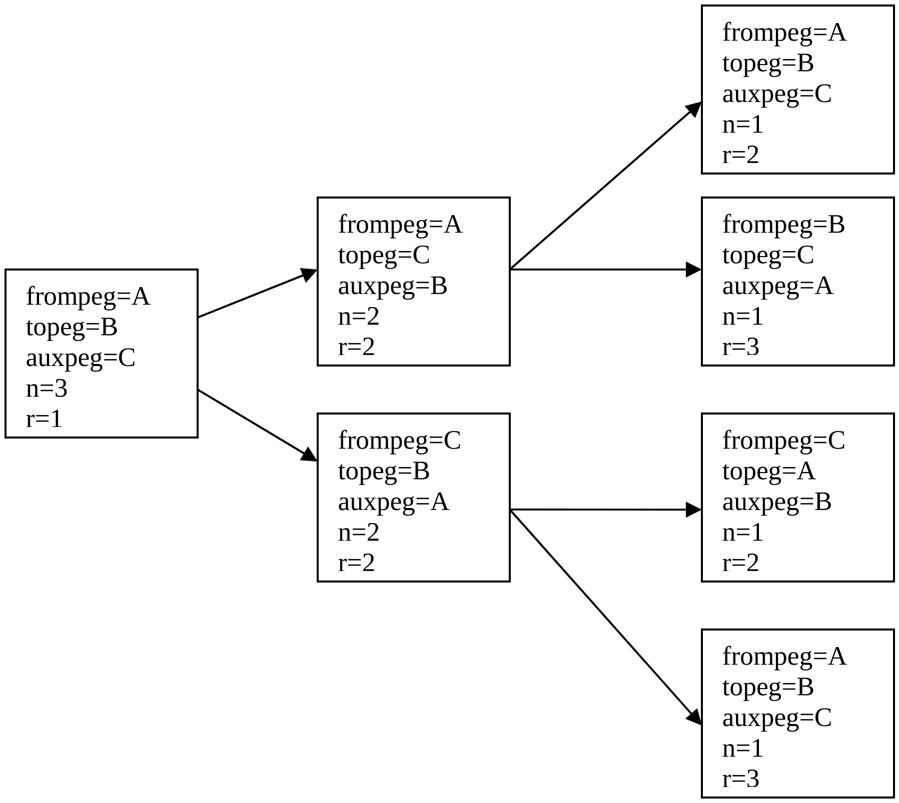
Your solution should print the moves in the format:

Move <disk no> from <pegno> to <pegno>

b) Convert the code into a non-recursive code by using stacks.

Hint: maintain a stack which has elements which are structures. The members of the structures are the fields, n, frompeg, topeg and auxpeg, to indicate the arguments defined above. Apart from these also note that there are three distinct points to which the function at any stage jumps to: to the main calling function or to the two subsequent recursive calls. Indicate this return addresses by labels: label1, label 2 and label3. Also maintain a start label and use goto statements to transfer the flow of the program. ***(You may or may not use the hint, but what is compulsory is the use of stacks).***

The following figure shows the flow of the program for n=3, frompeg=A, topeg=B and auxpeg=C. Here r denotes the return address.



c) Can you reduce the unnecessary stacking activities? Compare the run time of your codes with that of the recursive code.