26. When a parameter is passed to a subroutine by reference (i.e., not by value),
   a. the parameter can be put in an address register
   b. the address of the parameter can be put in an address register
   c. the address of the parameter can be pushed on the stack
   d. the parameter can be pushed on the stack
   e. parts a and d are correct
   f. parts b and c are correct CORRECT – you pass an address

27. Consider the following code:

   
   ```assembly
   MOVE.W X,-(A7)    Push X
   MOVE.L Y,-(A7)    Push Y
   BSR    PQR        Call PQR
   Clean_up        Clean up the stack
   ```

   a. Why do you have to clean up the stack after returning from the subroutine?
   b. What code would you use to clean up the stack?
   c. Draw a memory map of the stack immediately before executing the RTS in the subroutine PQR.

   a. The stack should be balanced in the sense that after you modify it, you should eventually reverse the modification. If a subroutine builds on top of the stack, you should restore the stack before returning.

   b. If you push a word and a longword on the stack, you move the stack pointer up by 6 bytes; that is $[A7] \leftarrow [A7] - 6$. You can undo this by adding 6 to the stack pointer with LEA 6 (A7), A7.

   c. Assume the stack is, say, $1000$ before this code is executed. The first `MOVE`, moves the stack up by 2 bytes to $00FF$. The second instruction moves the stack pointer up by 4 bytes to $00FFA$. The subroutine call saves the 4-byte return address on the stack and pushes up the stack pointer to $00FF6

   ```
   0FF6  Ret   Return address pushed on stack as a 16-bit word
   0FFA  YYY   Y pushed on stack as a 32-bit word
   OFFE  XX    X pushed on stack as a 16-bit word
   1000   0000  initial stack
   ```