**A code-generation algorithm:**

The algorithm takes as input a sequence of three-address statements constituting a basic block. For each three-address statement of the form x : = y op z, perform the following actions:

1.   Invoke a function getreg to determine the location L where the result of the computation y op z should be stored.

2.     Consult the address descriptor for y to determine y’, the current location of y. Prefer the register for y’ if the value of y is currently both in memory and a register. If the value of y is not already in L, generate the instruction MOV y’ , L to place a copy of y in L.

3.    Generate the instruction OP z’ , L where z’ is a current location of z. Prefer a register to a memory location if z is in both. Update the address descriptor of x to indicate that x is in location L. If x is in L, update its descriptor and remove x from all other descriptors.

4. If the current values of y or z have no next uses, are not live on exit from the block, and are in registers, alter the register descriptor to indicate that, after execution of x : = y op z , those registers will no longer contain y or z

**Generating Code for Assignment Statements:**

• The assignment d : = (a-b) + (a-c) + (a-c) might be translated into the following three-address code sequence:

Code sequence for the example is:

