## SHA-1 pseudocode

```
Note 1: All variables are unsigned 32-bit quantities and wrap modulo 2^32 when
calculating, except for
        ml, the message length, which is a 64-bit quantity, and
        hh, the message digest, which is a 160-bit quantity.
Note 2: All constants in this pseudo code are in big endian.

Initialize variables:

h0=0x67452301, h1=0xEFCDAB89, h2=0x98BADCFE, h3=0x10325476, h4=0xC3D2E1F0

ml = message length in bits (always a multiple of the number of bits in a
character).

Pre-processing:
append the bit '1' to the message e.g. by adding 0x80 if message length is a
multiple of 8 bits.
append 0 ≤ k < 512 bits '0', such that the resulting message length in bits
   is congruent to −64 ≡ 448 (mod 512)
append ml, the original message length, as a 64-bit big-endian integer.
   Thus, the total length is a multiple of 512 bits.

Process the message in successive 512-bit chunks:
break message into 512-bit chunks
for each chunk
    break chunk into sixteen 32-bit big-endian words w[i], 0 ≤ i ≤ 15

    Extend the sixteen 32-bit words into eighty 32-bit words:
    for i from 16 to 79
        w[i] = (w[i-3] xor w[i-8] xor w[i-14] xor w[i-16]) leftrotate 1

    Initialize hash value for this chunk:
    a = h0, b = h1, c = h2, d = h3, e = h4

    Main loop:[3][54]
    for i from 0 to 79
        if 0 ≤ i ≤ 19 then
            f = (b and c) or ((not b) and d)
            k = 0x5A827999
        else if 20 ≤ i ≤ 39
            f = b xor c xor d
            k = 0x6ED9EBA1
        else if 40 ≤ i ≤ 59
            f = (b and c) or (b and d) or (c and d)
            k = 0x8F1BBCDC
        else if 60 ≤ i ≤ 79
            f = b xor c xor d
            k = 0xCA62C1D6

        temp = (a leftrotate 5) + f + e + k + w[i]
        e = d
        d = c
        c = b leftrotate 30
        b = a
        a = temp

    Add this chunk's hash to result so far:
    h0 = h0 + a, h1 = h1 + b , h2 = h2 + c, h3 = h3 + d, h4 = h4 + e

Produce the final hash value (big-endian) as a 160-bit number:
hh = (h0 leftshift 128) or (h1 leftshift 96) or (h2 leftshift 64) or (h3
leftshift 32) or h4
```