program for subkey generation in CMAC, it states that the block cipher is applied to the block that consists entirely of 0 bits. The first subkey is derived from the resulting string by a left shift of one bit and, conditionally, by XORing a constant that depends on the block size. The second subkey is derived in the same manner from the first subkey. a. What constants are needed for block sizes of 64 and 128 bits? b. How the left shift and XOR accomplishes the desired result.

def left\_shift(b):

carry = 0

res = bytearray(len(b))

for i in range(len(b)-1, -1, -1):

new\_carry = (b[i] & 0x80) >> 7

res[i] = ((b[i] << 1) & 0xFF) | carry

carry = new\_carry

return bytes(res), carry

def generate\_subkeys(block\_size):

if block\_size == 64:

const = 0x1B

n = 8

elif block\_size == 128:

const = 0x87

n = 16

L = bytes(n)

K1, carry = left\_shift(L)

if carry:

K1 = bytearray(K1)

K1[-1] ^= const

K1 = bytes(K1)

K2, carry = left\_shift(K1)

if carry:

K2 = bytearray(K2)

K2[-1] ^= const

K2 = bytes(K2)

return K1, K2

K1\_64, K2\_64 = generate\_subkeys(64)

K1\_128, K2\_128 = generate\_subkeys(128)

print("64-bit Subkeys:")

print("K1:", K1\_64.hex().upper())

print("K2:", K2\_64.hex().upper())

print("128-bit Subkeys:")

print("K1:", K1\_128.hex().upper())

print("K2:", K2\_128.hex().upper())