program for ECB, CBC, and CFB modes, the plaintext must be a sequence of one or more complete data blocks (or, for CFB mode, data segments). In other words, for these three modes, the total number of bits in the plaintext must be a positive multiple of the block (or segment) size. One common method of padding, if needed, consists of a 1 bit followed by as few zero bits, possibly none, as are necessary to complete the final block. It is considered good practice for the sender to pad every message, including messages in which the final message block is already complete. What is the motivation for including a padding block when padding is not needed?

def pad\_message(msg, block\_size):

msg\_bits = ''.join(format(ord(c), '08b') for c in msg)

pad\_len = block\_size - (len(msg\_bits) % block\_size)

if pad\_len == 0:

pad\_len = block\_size

padded = msg\_bits + '1' + '0' \* (pad\_len - 1)

return padded

def split\_blocks(bits, block\_size):

return [bits[i:i+block\_size] for i in range(0, len(bits), block\_size)]

msg = "HELLO"

block\_size = 8

padded\_bits = pad\_message(msg, block\_size)

blocks = split\_blocks(padded\_bits, block\_size)

print("Original message:", msg)

print("Padded binary:", padded\_bits)

print("Blocks:", blocks)

print("Motivation: Padding even when full prevents ambiguity in removing padding during decryption.") 