CPA code explanation:

- Here we have Encr_cpa() to encrypt the message and it returns cipher text
- Decr_cpa() which decrypts the cipher text and returns plain text
- Encr_cpa() receives message m , key k ,and Initialization vector r as inputs
- First it will apply block cipher on (k,r)
- Next we divide m into fixed length sized blocks of BLOCK_SIZE=16
- We apply block cipher on every block and prev block ciphers output ,means prev PRF's output and current block and gets ciphertext for that block
- This becomes input for the next block cipher operation.
- At the end we return concatenated cipher text
- Decr_cpa() receives cipher text , key , r and returns plain text by applying same encryption algorithm but instead of plain text it will put cipher text
- At the end we print decrypted text and true/false which indicates whether decrypted text is matched with original message or not .

```
def Encr_cpa(m, k, r):
    f = PRF(k, r)  # Fk(r)
    msg_len = len(m)
    rem = msg_len % BLOCK_SIZE
    total_blocks = msg_len//BLOCK_SIZE
    cipher_blocks = list()
    for block in range(0, msg_len-rem, BLOCK_SIZE):
        msg_bits = m[block:block+BLOCK_SIZE]
        cipher_text = ""
        for i in range(len(msg_bits)):
            xor_val = int(msg_bits[i]) ^ int(f[i])
            cipher_text += str(xor_val)
        cipher_blocks.append(cipher_text)
        f = PRF(k, f)
    if rem != 0:
        cipher_text = ""
        msg_bits = m[-1*rem:]
        for i in range(rem):
            xor_val = int(msg_bits[i]) ^ int(f[i])
            cipher_text += str(xor_val)
        cipher_blocks.append(cipher_text)
    final_cipher = ""
```

```
    for text in cipher_blocks:
        final_cipher += text
    return final_cipher


def Decr_cpa(m, k, r):
    # sending cipher to text same algo instead of plain text
    plain_text = Encr_cpa(m, k, r)
    return plain_text
```