# Homework : Bitwise rotate and XOR calculate

Cryptography plays a crucial role in the field of computer science. This assignment allow you to practice implementing a simple encryption scheme.

## Encryption

1. Input a continuous sequence of characters.
2. Convert each character's ASCII code into 8-bit binary

   (ASCII is 7-bit, but we store it in 8 bits by adding a leading 0).
3. Perform a bitwise rotate-right by 3 bits on the 8-bit value.
4. Apply XOR with the key.
5. Convert the resulting 8-bit binary into hexadecimal,

   and output the hexadecimal ciphertext.

## Decryption

1. Input the hexadecimal ciphertext into the program.
2. Convert each pair of hex-digits back into an 8-bit binary value.
3. Perform a bitwise rotate-left by 3 bits on the 8-bit value.
4. Apply XOR with the same key.

5. Convert the resulting 8-bit binary to decimal, and print the corresponding ASCII character, which is the original plaintext.

   HINT: KEY:0X6D ,right_rotate_3bits(x)= (x>>3 | x<<5)

## Ex:

cat plaintext.txt

```
Hello World !
```

gcc encrypt.c -o encrypt

./encrypt<plaintext.txt>ciphertext.txt

cat ciphertext.txt

```
64C1E0E08069878023E0E16949
```

gcc decrypt.c -o decrypt

./decrypt<ciphertext.txt>PlainText.txt

cat PlainText.txt

```
Hello World !
```

Please submit two .c files and include a screenshot showing all of the processing steps. You may test with any inputs you like.

# 加密流程（Encryption）

1. 連續輸入一串字元。
2. 將每個字元的 ASCII 碼轉換為 8-bit 二進位值

   （ASCII 本為 7-bit，但在電腦中以 8-bit 儲存，最高位補 0）。
3. 對此 8-bit 值進行 bitwise 向右旋轉（rotate-right）3 bit。
4. 使用指定的 key 進行 XOR 運算。
5. 將 XOR 後得到的 8-bit 二進位 轉換成 十六進位（hexadecimal），

   並輸出作為密文。

# 解密流程（Decryption）

1. 將十六進位格式的密文輸入程式。
2. 將每兩個十六進位字元還原成 8-bit 二進位值。
3. 對此 8-bit 值進行 bitwise 向左旋轉（rotate-left）3 bit。
4. 使用相同的 key 進行 XOR 運算。
5. 將 XOR 後得到的 8-bit 二進位 轉換成 十進位（decimal），

   並輸出該十進位對應的 ASCII 字元（原始明文）。

（範例如上）