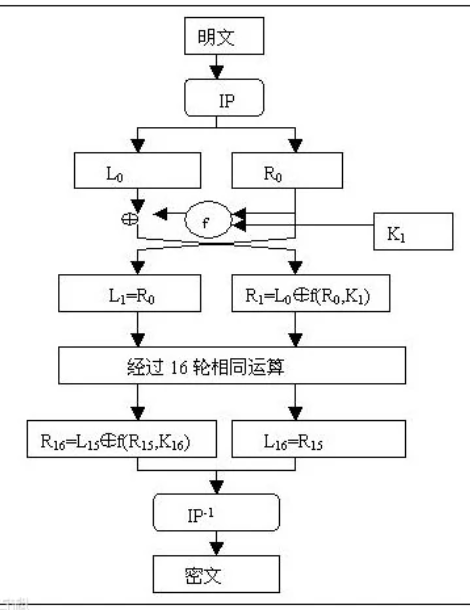
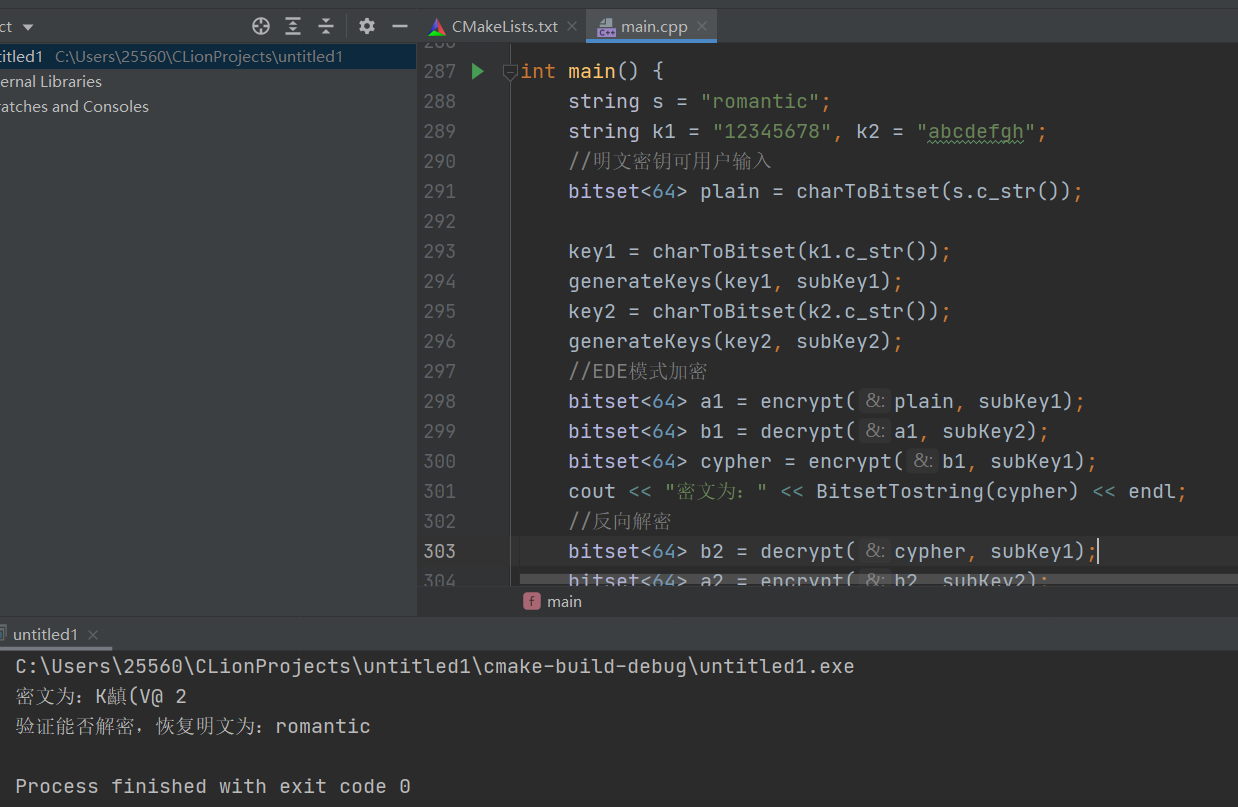
# 使用两个密钥的三重DES原理及实现

1. DES原理图：



1. 测试函数及运行结果：



1. 程序分析：可以看到明文设置为”romantic”之后，使用k1、k2两个密钥经EDE模式加密之后产生的密文为结果第一行的字符串（未输出为ASCII看起来像乱码）。再通过DED解密可恢复明文，说明函数编写成功。
2. 主要功能代码：

* 每次迭代只处理右半部分的f函数

bitset<32> f(bitset<32> R, bitset<48> k) {  
 bitset<48> expandR;  
 // 第一步：扩展置换，32 -> 48  
 for (int i = 0; i < 48; ++i)  
 expandR[47 - i] = R[32 - E[i]];  
 // 第二步：异或  
 expandR = expandR ^ k;  
 // 第三步：查找S\_BOX置换表  
 bitset<32> output;  
 int x = 0;  
 for (int i = 0; i < 48; i = i + 6) {  
 int row = expandR[47 - i] \* 2 + expandR[47 - i - 5];  
 int col = expandR[47 - i - 1] \* 8 + expandR[47 - i - 2] \* 4 + expandR[47 - i - 3] \* 2 + expandR[47 - i - 4];  
 int num = S\_BOX[i / 6][row][col];  
 bitset<4> binary(num);  
 output[31 - x] = binary[3];  
 output[31 - x - 1] = binary[2];  
 output[31 - x - 2] = binary[1];  
 output[31 - x - 3] = binary[0];  
 x += 4;  
 }  
 // 第四步：P-置换，32 -> 32  
 bitset<32> tmp = output;  
 for (int i = 0; i < 32; ++i)  
 output[31 - i] = tmp[32 - P[i]];  
 return output;  
}

* 子密钥产生器

void generateKeys(bitset<64> key, bitset<48> subKey[16]) {

bitset<56> realKey;

bitset<28> left;

bitset<28> right;

bitset<48> compressKey;

// 去掉奇偶标记位，将64位密钥变成56位

for (int i = 0; i < 56; ++i)

realKey[55 - i] = key[64 - PC\_1[i]];

// 生成子密钥，保存在 subKeys[16] 中

for (int round = 0; round < 16; ++round) {

// 前28位与后28位

for (int i = 28; i < 56; ++i)

left[i - 28] = realKey[i];

for (int i = 0; i < 28; ++i)

right[i] = realKey[i];

// 左移

left = leftShift(left, shiftBits[round]);

right = leftShift(right, shiftBits[round]);

// 压缩置换，由56位得到48位子密钥

for (int i = 28; i < 56; ++i)

realKey[i] = left[i - 28];

for (int i = 0; i < 28; ++i)

realKey[i] = right[i];

for (int i = 0; i < 48; ++i)

compressKey[47 - i] = realKey[56 - PC\_2[i]];

subKey[round] = compressKey;

}

}

1. 源代码：见main.cpp
2. 编程环境：Clion+WIN10