Compilation of C Program Questions — **Ultimate Section**

by Jaser Li (HDU) Sept. 2023

* 多重解密：精密、逻辑、高效

背景

精密，不容错误；逻辑，严丝合缝；高效，深合底层。Precise, Logistic, Efficient. 此乃对伟大的位运算的赞歌。快捷有力，其应用与效果令人惊叹。值得一试，无需等待，现在，它就在你的面前。对应用感到疑惑的话，为何不去检索资料，看看它的效果呢？

题目描述

现在，你是匈牙利文明的双料高级特工，执行来自帝国领袖凌的命令，将要对奥斯曼文明实施陷害总督任务。最高领导人将给你发来一封密函和一封附件，这封密函已使用你对应的密钥加密，必须使用你手上的密钥解密。

密钥内容：

unsigned char password[70]={71,114,101,97,116,32,105,100,101,97,108,32,98,117,

116,32,116,104,114,111,117,103,104,32,115,101,108,102,108,101,115,115,32,115,116,114,117,103,103,108,101,32,97,110,100,32,115,97,99,114,105,102,105,99,

101,32,116,111,32,97,99,104,105,101,118,101,46};

**第一重解密：对密函按位异或**

密函发来前，每位数字都曾使用你手上的密钥作按位异或运算，现在，重复一次按位异或运算就可以得到原文。（为什么呢？）

**第二重解密：对附件循环移位**

附件发来前特别作循环位移处理。你需要将附件的每一个数据向左循环位移n位，n的值将被给出。

所谓循环位移：例如给出的数为1001，要求循环左移二位。

1001 ->

0011 ->

0110.

结果为0110, 也即十进制数9在四位二进制数表示下的循环左移二位结果是6.

这是在数据位数限定为4位的前提下得到的结果。如果限定数据位数为8位，那么其左移结果如下

00001001 ->

00010010 ->

00100100.

结果为00100100. 也即十进制数9在八位二进制数表示下的循环左移二位结果是36.

在本次任务中，数据位数限定为8位，为确保位移过程中的最高位统一，请声明变量为unsigned char类型。

**第三重解密：按位与**

为了保证任何一封密函单独被敌人截获都不会导致泄密，每一封密函都随附一封附件，将解密后的附件和解密后的密函按位与才能得到真正的内容。将解密后密函与解密后附件中的每位数字按位与，获取来自帝国的指令。

Format

Input

输入将包含三行。第一行是**密函**，第二行是**附件**，第三行是**循环位移位数**。

附件、密函和密钥不一定等长！（保证密函长度是最短的）当长度不匹配时，以密函长度len为准，附件与密钥取前len位有效。

Output

解密后的原文。

Samples

输入数据 1

9 23 19 132 6 192 6 27 10 15 13 94 29 14 3 69 4 17 29 18 73 18 24 78

201 172 222 239 206 5 254 237 221 207 172 20 252 61 206 188 21 175 253 254 68 190 126 215

3

输出数据 1

78 101 118 101 114 32 103 111 110 110 97 32 103 105 118 101 32 121 111 117 32 117 112 46

Limitation

1s, 1024KiB for each test case. len≤70, −8≤n≤8.