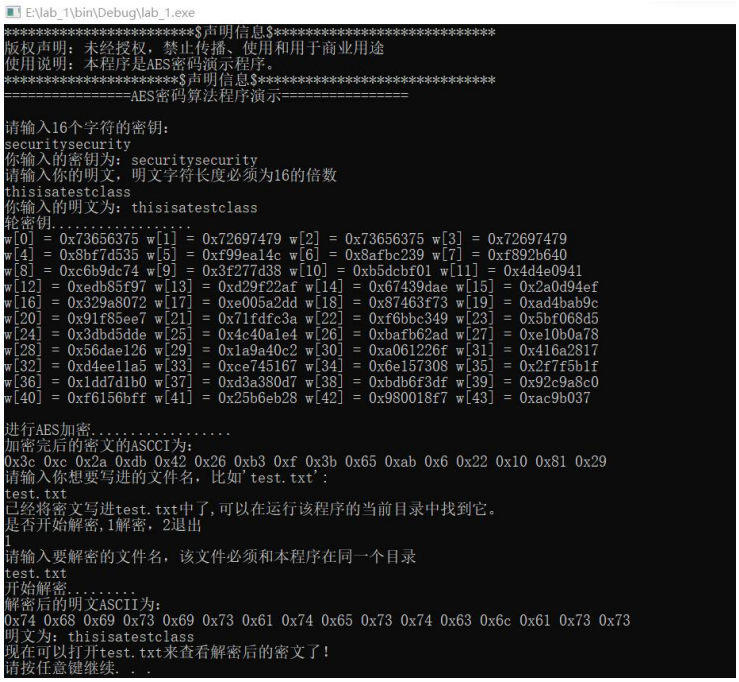
实验一 AES 密码算法

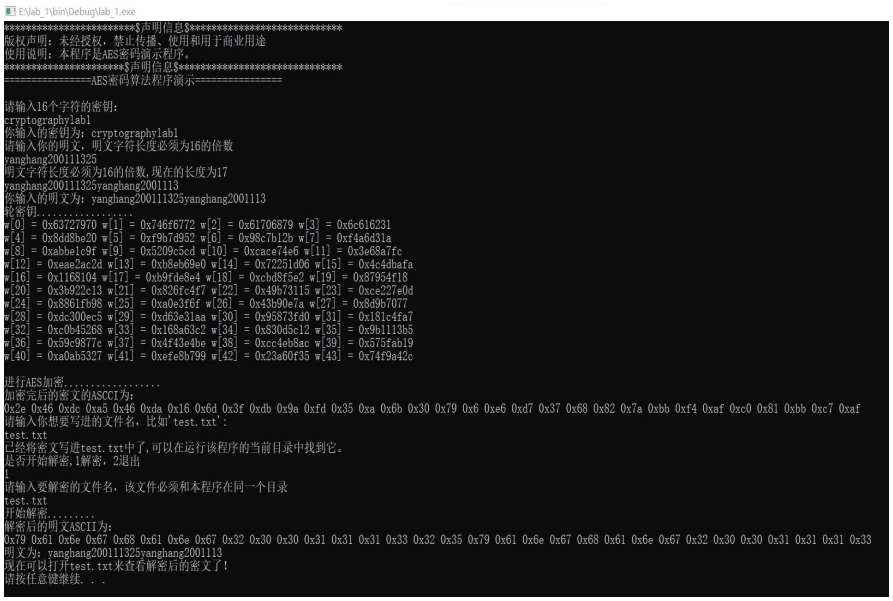
姓名：_____杨行_____学号：____2001111325_____

一、 运行截图

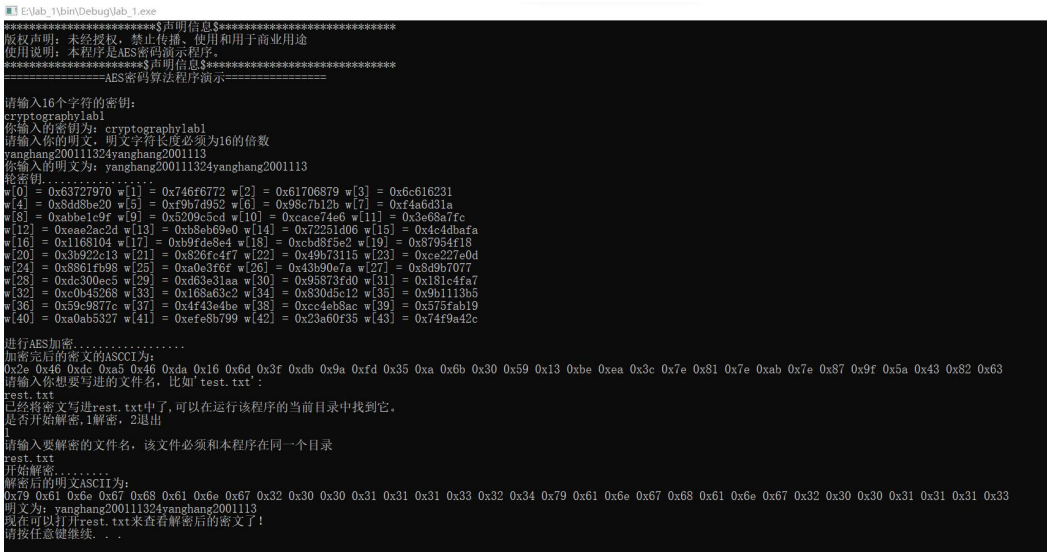
第一组：明文为 thisisatestclass,密钥为 securitysecurity



第二组：姓名拼音+学号（16 位不够补为 32 位） 密钥为：cryptographylab1



第三组：姓名拼音+（学号-1）（16 位不够补为 32 位）密钥为：cryptographylab1

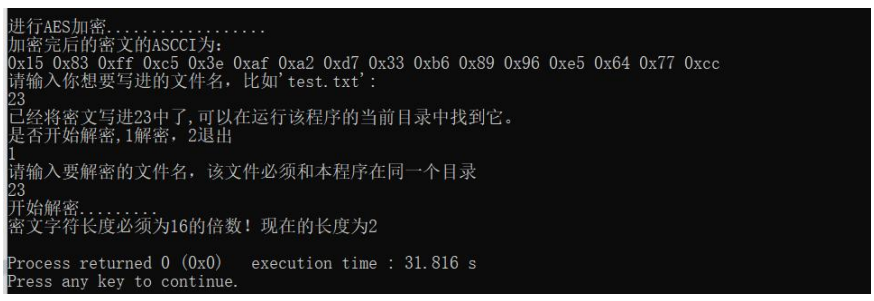


二、 实验过程中遇到的问题有哪些？你是怎么解决的。

答：实验过程主要遇到的问题大多在于对理论的不熟悉导致根据模板代码补全时存在一定问题，通过回顾知识点得以解决。实验中遇到的一个有意思的问题是原有的 readStrFromFile 函数读取文件中有一句读的结束判断如下：

```
for(i = 0; i < MAXLEN && (str[i] = getc(fp)) != EOF ; i++);
```

由于 str 定义为 char，因为文本文件中存储的是 ASCII 码，而 ASCII 码中 FF 代表空值（blank），也就是说，在语句“(str[i]=getc(fp))!=EOF”中，当读取的字符为 0xFF（空格）时，“getc(fp)”的值由 0x000000FF 转换为 char 类型（ 0xFF）；而在执行语句“str[i]！=EOF”时，字符与整数比较，str[i]被转换为 0xFFFFFFFF，条件成立，遇到空格字符时就退出。导致密文中存在空格则无法解密。如下图：



密文第三位出现 0xFF,无法解密

```
for(i = 0; i < MAXLEN ; i++){
        str[i] = getc(fp);
       if(feof(fp)){
        break;
       }

      // printf("%c",str[i]);

};

rewind(fp);
```

改为用 feof 判断文本结尾可以有效解决密文中间出现 0xff 现象。

但如果密文结尾出现 0xff 现象，则依旧无法判断，这种情况想到的解决办法是在文本追加末尾字符保存明文长度，依此保证读入字符个数正确，或读取时自动判断是否为 16 的倍数，但两者都存在对错误文件解密的可能，故不为采用。

三、 如果不用 lab1-aes.c 代码框架或者实现了 CBC 模式，请说明。