专业课程设计报告

（ 2022/ 2023学年 第 一 学期）

题 目： U盘加密系统

|  |  |
| --- | --- |
| **专 业** | **信息安全** |
| **学 生 姓 名** | **任远哲** |
| **班 级 学 号** | **B190307-B19031614** |
| **指 导 教 师** | **张洁** |
| **指 导 单 位** | **计算机学院、软件学院、网络空间安全学院** |
| **日 期** | **2022.10.24-2022.11.06** |

|  |  |  |
| --- | --- | --- |
| **课程目标** | **评价准则** | **计分（每项10分）** |
| 课程目标1：通过课程设计，培养学生综合应用信息安全、计算机技术等领域专业知识的技能。（20分） | 1、能够掌握信息安全的相关基础知识，并能够针对求解的工程问题，收集资料进行合理的分析与设计。 |  |
| 2、从软件的分析、设计到编制调试，结合计算机网络理论知识、编程语言以及程序设计的方法，能够解决一个和信息安全相关的问题。 |  |
| 课程目标2：解决信息安全领域复杂工程问题的实践创新能力。（20分） | 3、通过调研，能够选择合适的程序设计语言与编程开发平台，对求解的工程问题进行编程实现。 |  |
| 4、具备一定的人机交互设计意识，人机交互设计合理、友好，操作简便。 |  |
| 课程目标3：文献调研与资料收集能力，问题发现、研究、分析与解决能力。（10分） | 5、具备一定自学能力与探索创新意识，能够充分利用教科书及其资源（如网络等）自学新知识与新技能。 |  |
| 课程目标4：培养工程工具运用能力，能够利用仿真软件或实验系统对信息安全系统进行模拟和预测，并理解仿真软件或实验硬件的局限性。（20分） | 6、能够结合计算机软硬件资源，合理选用算法、数据结构、数据存储方式等技术手段，理解相关算法，对求解的工程问题进行有效建模和求解。 |  |
| 7、掌握调试方法与工具，对程序开发过程中出现的问题进行分析、跟踪与调试，并能够进行充分测试。 |  |
| 课程目标5：分组完成一次项目设计与开发的全过程，组内成员通过讨论和交流解决课程设计中的难题，能在实验报告中准确阐述课程设计的内容，能够清晰陈述观点和回答问题。（30分） | 8、组内成员之间有一定的团队合作，互通有无。 |  |
| 9、具备一定的语言表达能力与文字处理能力，能够结合复杂工程问题撰写报告，报告内容和实验数据详实，格式规范。 |
| 10、能够正确、完整地回答指导教师关于课题的问询，反映其对课题内容，以及相关的工程基础知识具有较好的理解和掌握。 |
| 专业课程设计能力测评总分 | |  |
| **指导教师： 年 月 日** | | |
| **备注：加上平时成绩之后换算的总评成绩及等级（优、良、中、及格等）：** | | |

**U盘加密系统**

**一、课题内容和要求**

在学习了程序设计和信息安全专业基础课程的基础上，综合运用所学的方法和理论，编写对u盘进行完全加密或者部分加密的程序，从软件的分析、设计到编制调试，要求学生把信息安全、编程语言以及程序设计的方法等结合起来，并通过课程设计提高学生的查阅资料、自学和独立分析问题与解决问题的能力，加深对理论知识的理解，利用仿真软件或实验系统对信息安全系统进行模拟和预测，验证并掌握信息安全学科中重要的基础理论与方法，为今后从事相关方面的工作奠定基础。

题目的具体要求如下：

1.对u盘进行部分或者全部加密；

2.采用对称加密算法进行u盘数据加密；

3.对称加密算法的密钥要用口令或者公钥加密；

4.查阅相关资料，深入理解分组加密AES和公钥加密RSA的原理；

5.程序具有图形化用户界面，输出美观；

6.可根据自己能力，在完成以上基本要求后，对程序功能进行适当扩充；

7.撰写报告，对所采用的算法、程序结构和主要函数过程以及关键变量进行详细的说明；对程序的调试过程所遇到的问题进行回顾和分析，对测试和运行结果进行分析；总结软件设计和实习的经验和体会，进一步改进的设想；

8.提供关键程序的清单、源程序及可执行文件和相关的软件说明。

**二、需求分析和总体设计**

2.1本课题的主要功能包括：

（1）输入路径和对称密钥，使用AES算法加密U盘（文件夹）

（2）输入路径和对称密钥，使用AES算法解密（1）中加密后U盘（文件夹）

（3）输入RSA公钥e和n，使用RSA算法加密（1）中的对称密钥

（4）输入RSA私钥d和n，使用RSA算法解密（3）中加密的对称密钥

（5）自动生成RSA参数（e，d，n）

**本人是组长，完成gui界面的编写以及功能(1),(2)，并整合所有代码**

绘制用例图，如图1所示：

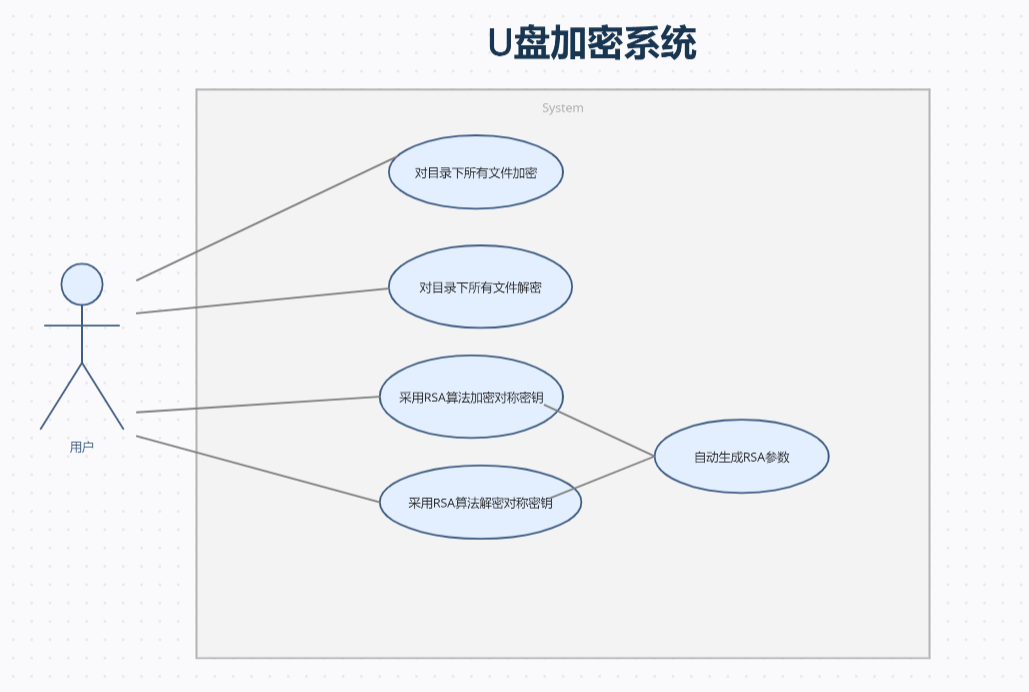


图1用例图

2.2 本课题的数据表单设计

本加密系统使用的一些参数：

（1）filepaths：存储根目录及其子目录下所有文件的路径

（2）AESE.blk = blk ：每一分组的明文list[int]

AESE.key = key ：每一分组的密钥list[int]

AESE.Nr = Nr ：对每一分组加密的轮次

（3） e：RSA的公钥

d：RSA的私钥

n：RSA的模

2.3 本课题的体系结构设计

因为功能并不复杂，所以本系统采用“调用和返回体系结构”风格中的“主程序/子程序体系结构”。 这种的程序结构将功能分解成一个控制层次，其中主程序调用一组程序构件，这些程序构件又去调用别的程序构件。编程是基于面向过程的思想，加密算法模块:对文件AES和对字符串RSA 由自己编写，GUI界面则调用了Python中的Tkinter库。

2.4本课题中的主要交互行为建模，绘制序列图，如图2所示：

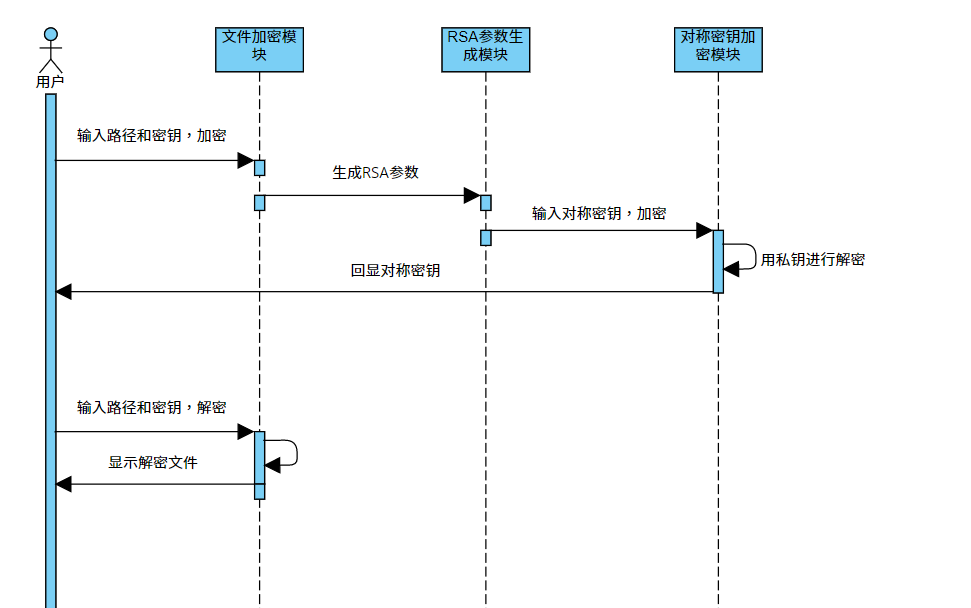


图2序列图

首先将一个文件读取成字节流，再按16字节为一个分组进行划分，最后对每个分组分别调用AES算法加密。RSA算法是公钥加密算法，我们首先生成RSA的公私钥等参数，这里我们加密的对象是字符串（AES算法中的对称密钥），将每个字符转化成其ascill码（int型）以用RSA算法分别加密。

2.5 本课题的主要功能界面设计

本U盘加密系统总共设计了5个子页面。页面1为“Encrypt package via AES”，即允许用户输入路径和对称密钥，使用AES算法加密U盘（文件夹）；页面2为“Decrypt package via AES”，即允许用户输入路径和对称密钥，使用AES算法解密（1）中加密后U盘（文件夹）；页面3为“Encrypt AES Key via RSA”，即允许用户输入RSA私钥d和n，使用RSA算法解密（3）中加密的对称密钥；页面4为“Decrypt AES Key via RSA”，即允许用户输入RSA私钥d和n，使用RSA算法解密（3）中加密的对称密钥；页面5为 “RSA parameters”，即允许用户自动生成RSA参数（e，d，n）。

因为篇幅限制，下面仅展示页面1的图形化界面，如图3所示：

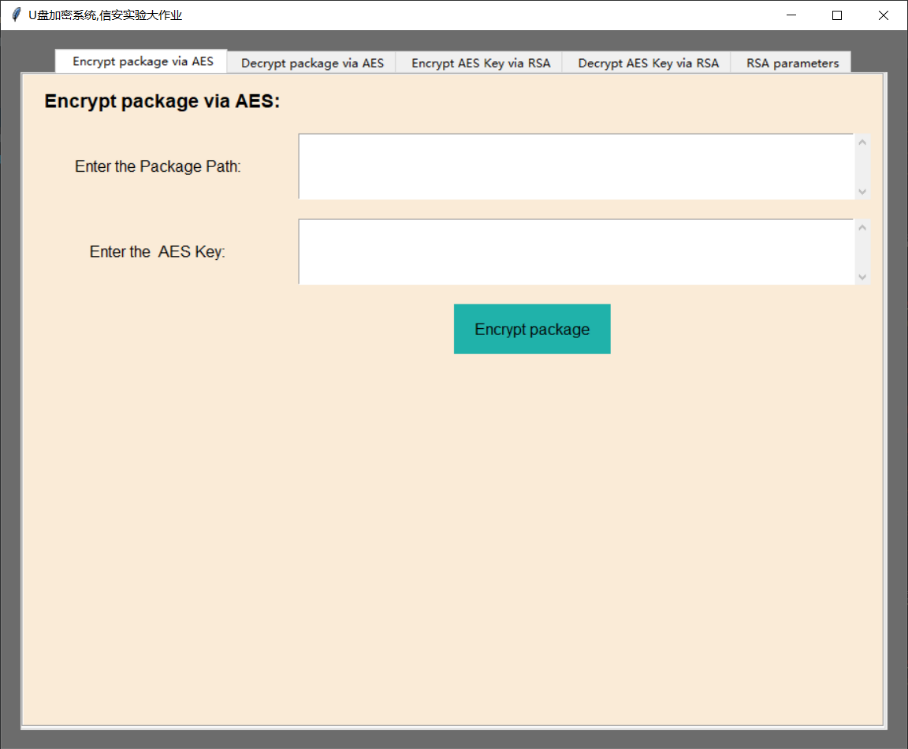


图3界面图

**三、相关功能模块详细设计**

该系统主要分为三个模块：gui.py，aes\_file.py和rsa.py

**（1）gui.py** ：GUI界面则调用了Python中的Tkinter库进行可视化，分为五个界面。每个界面获取ScrolledText框中输入的参数，通过Button键调用aes\_file.py和rsa.py中的核心算法进行处理，并回显计算结果。

**（2）aes\_file.py** ：该模块负责实现对文件夹中的文件进行AES加密的操作。 首先通过函数 all\_files\_path(rootDir) 对根目录rootDir下的所有文件进行递归遍历，然后再依次以二进制的形式读取这些文件。每一个文件按16字节为一个分组进行划分，然后进行分组加密或解密。

AES 密码学中的高级加密标准（Advanced Encryption Standard，AES），又称Rijndael加密法，是美国联邦政府采用的一种区块加密标准。这个标准用来替代原先的DES（Data Encryption Standard），已经被多方分析且广为全世界所使用。

AES加解密算法流程

AES的区块长度固定为128位（16字节）。[密钥](https://so.csdn.net/so/search?q=%E5%AF%86%E9%92%A5&spm=1001.2101.3001.7020)长度则可以是128 bit，192 bit 或256位 bit，换算成字节长度，就是密码必须是 16个字节，24个字节，32个字节（在这个实验中我们选择16字节的版本）。

AES的加密模式有以下几种：电码本模式(ECB)，密码分组链接模式(CBC)，计算器模式(CTR)，密码反馈模式(CFB)，输出反馈模式(OFB)。在本次实验中我们选择电码本模式(ECB)，即将需要加密的消息按照块密码的块大小被分为数个块，并对每个块进行独立加密。

ECB工作模式要求输入明文长度必须是块（16 byte）长度的整数倍，因此信息必须填充至满足要求。AES支持的填充模式主要分为PKCS7和ZerosPadding。在本次实验中我们选择ZerosPadding填充模式，即全部填充0x00，无论缺多少全部填充0x00，已经是128bits倍数仍要填充一个全0的块。

AES加密算法涉及4种操作：字节替代（SubBytes）、行移位（ShiftRows）、列混淆（MixColumns）和轮密钥加（AddRoundKey），如下图4所示：

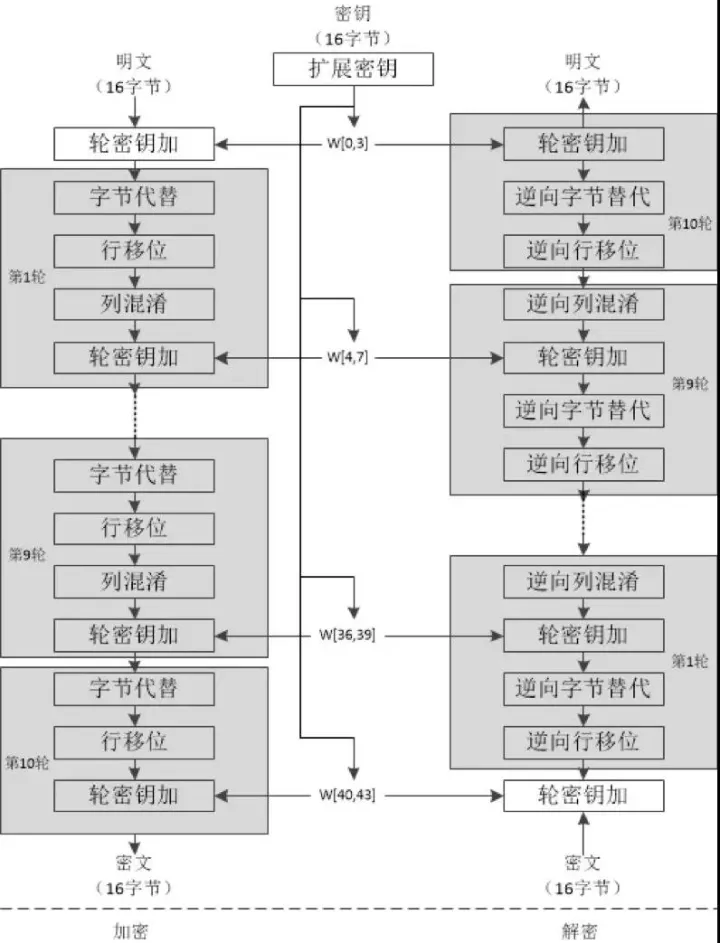


图4 AES流程图

（2.1）字节替代

字节代替的主要功能是通过S盒完成一个字节到另外一个字节的映射。AES定义了一个S盒和一个逆S盒，用于提供密码算法的混淆性。S和S-1分别为16x16的矩阵，完成一个8比特输入到8比特输出的映射，输入的高4-bit对应的值作为行标，低4-bit对应的值作为列标。

（2.2）行移位

行移位是一个4x4的矩阵内部字节之间的置换，用于提供算法的扩散性。

2.2.1) 正向行移位，如图5

正向行移位用于加密，其原理图如下。其中：第一行保持不变，第二行循环左移8比特，第三行循环左移16比特，第四行循环左移24比特。

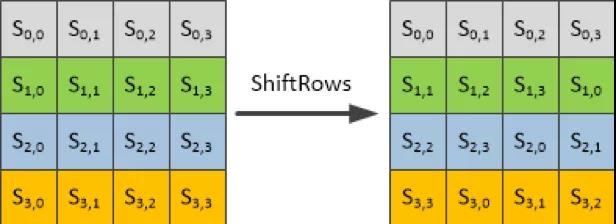


图5正向行移位图

2.2.2) 逆向行移位

　　逆向行移位即是相反的操作，即：第一行保持不变，第二行循环右移8比特，第三行循环右移16比特，第四行循环右移24比特。

（2.3）列混淆

2.3.1) 正向列混淆，如图6

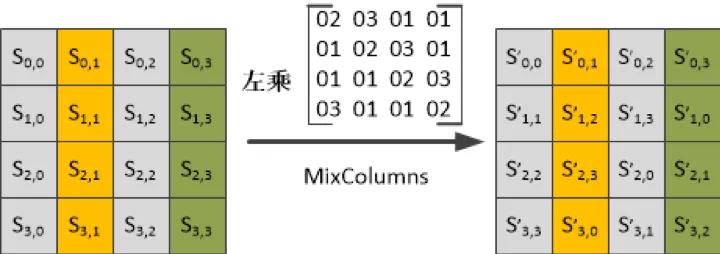


图6正向列混淆图

2.3.2) 逆向列混淆，如图7

逆向列混淆的原理图如下：

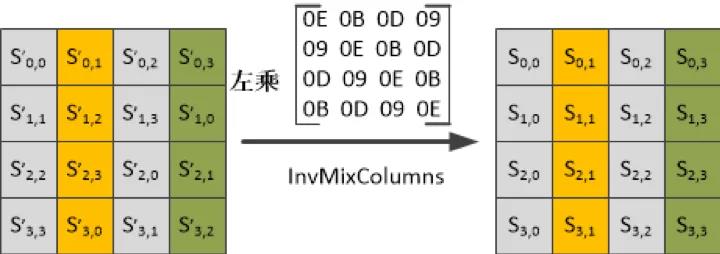


图7逆向列混淆图

可以看出说明两个矩阵互逆，经过一次逆向列混淆后即可恢复原文，如图8

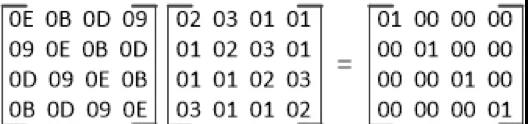


图8两个矩阵互逆图

根据矩阵的乘法，在列混淆的过程中，每个字节对应的值只与该列的4个值有关系。

假设某一列的值如下图9，运算过程如下：

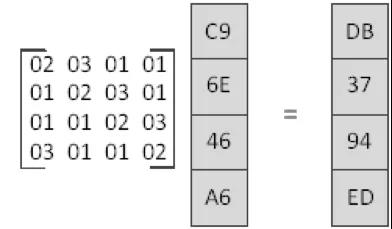


图9混淆运算图

（2.4）轮密钥加

这个操作相对简单，其依据的原理是“任何数和自身的异或结果为0。加密过程中，每轮的输入与轮子密钥异或一次；因此，解密时再异或上该轮的轮子密钥即可恢复。

密钥扩展：其复杂性是确保算法安全性的重要部分。当分组长度和密钥长度都是128位时，AES的加密算法共迭代10轮，需要10个子密钥。AES的密钥扩展的目的是将输入的128位密钥扩展成11个128位的子密钥。AES的密钥扩展算法是以字为一个基本单位（一个字为4个字节），刚好是密钥矩阵的一列。因此4个字（128位）密钥需要扩展成11个子密钥，共44个字。

**（3）rsa.py:** 该模块主要负责对一个字符串（AES对称密钥）进行RSA加密和解密。我们的做法是将待加密的字符串按其中每个字符的ascill码分别加密，中间用’,’隔开。其中公钥e，私钥d和参数n 可以由我们指定，也可以自动生成。

RSA加密是一种非对称加密。可以在不直接传递密钥的情况下，完成解密。这能够确保信息的安全性，避免了直接传递密钥所造成的被破解的风险。是由一对密钥来进行加解密的过程，分别称为公钥和私钥。RSA 的安全性依赖于分解两个大[素数](https://so.csdn.net/so/search?q=%E7%B4%A0%E6%95%B0&spm=1001.2101.3001.7020" \t "_blank)乘积的实际困难。其具体算法流程如下：

（3.1）选择两个质数p和q，算出他们的乘积n = p × q，算出对应的欧拉函数φ ( n )（利用性质φ ( n ) = φ ( p ) × φ ( q ) = ( p − 1 ) ( q − 1 )。

（3.2）选择一个e，使得e < φ ( n )并且e与φ ( n )互质。

（3.3）算出e的一个相对于φ ( n )的模反元素d。( e , n )为公钥，( d , n )为私钥，信息（明文）m长度小于n。

（3.4）加密：c =m^e (mod n)

（3.5）解密：m =c^d (mod n)。

**四、部分核心代码**

**4.1 GUI界面模块的代码**

4.1.1主框架绘制代码，设置标题和其下5个子页面，如图10所示：

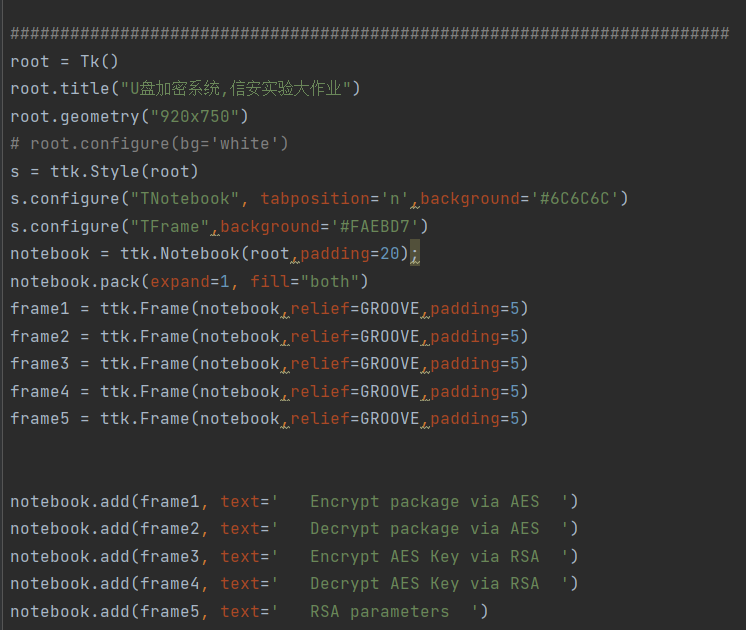


图10 主框架绘制代码图

4.1.2子页面1代码。输入框输入根目录和公钥并对其中所有文件进行加密，如图11：



图11 子页面1代码图

由于篇幅限制，子页面2-5的代码不展示

**4.2 递归遍历根目录所有文件模块，如图12所示**

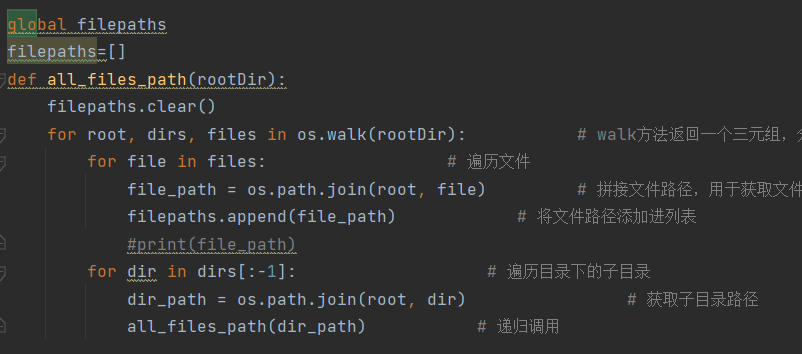


图12 递归遍历根目录代码图

**4.3 AES加密模块**

4.3.1 字节替代模块，如图13所示：

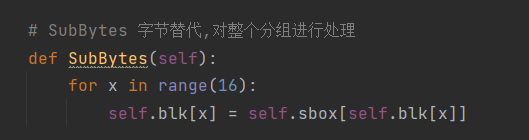


图13 字节替代代码图

4.3.2 行移位模块，如图14所示：

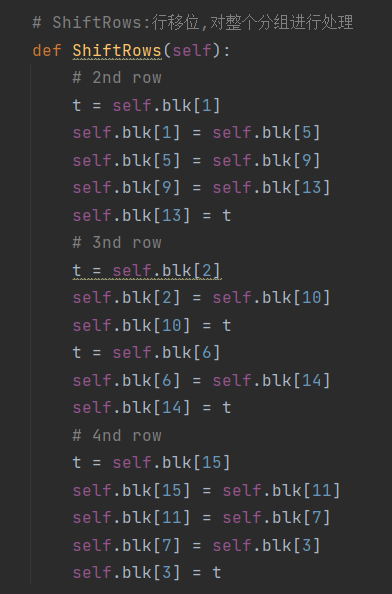


图14 行移位模块代码图

4.3.3 列混淆模块，如图15所示：

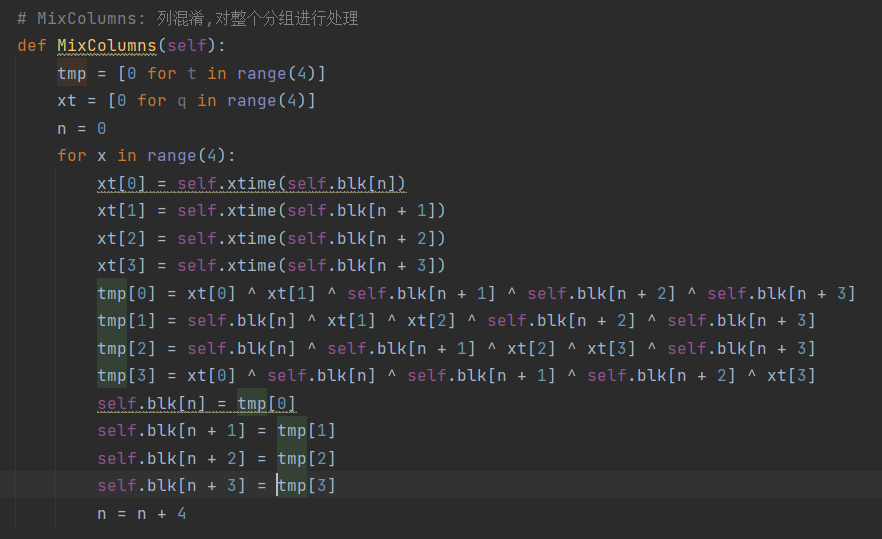


图15 列混淆模块代码图

4.3.4 轮密钥加模块，如图16所示：

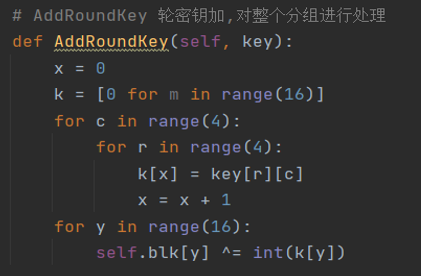


图16 轮密钥加模块代码图

4.3.5 密钥拓展模块，如图17所示：

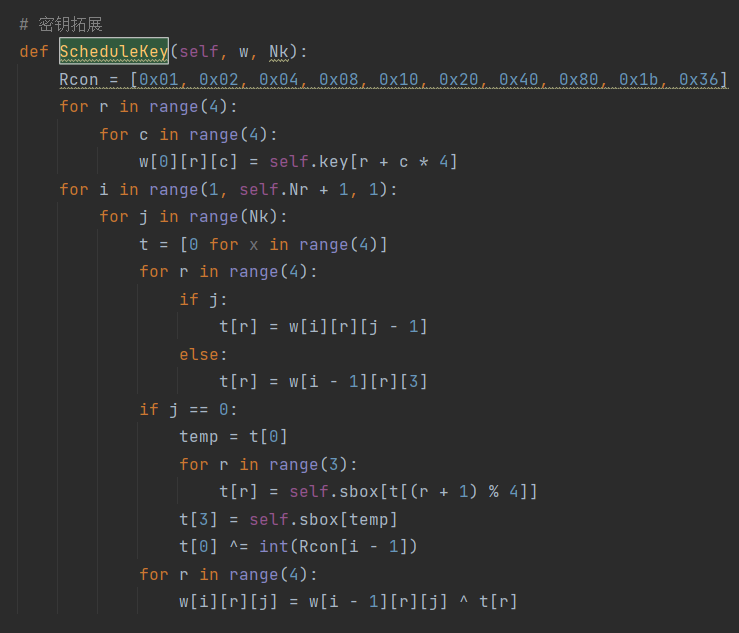


图17 密钥拓展模块代码图

由于篇幅限制，解密模块的代码不展示

**4.4 对AES密钥进行RSA加密模块**

4.4.1 RSA加密模块，如图18所示：



图18 RSA加密模块代码图

**五、软件测试及其结果分析**

**5.1单元测试（以递归枚举根路径下的所有文件为例）**

import os

filepaths=[]

def all\_files\_path(rootDir):

filepaths.clear()

for root, dirs, files in os.walk(rootDir):

for file in files: # 遍历文件

file\_path = os.path.join(root, file) # 拼接文件路径，用于获取文件绝对路径

filepaths.append(file\_path) # 将文件路径添加进列表

print(file\_path)

for dir in dirs[:-1]: # 遍历目录下的子目录

dir\_path = os.path.join(root, dir) # 获取子目录路径

all\_files\_path(dir\_path) # 递归调用

all\_files\_path('Z:\TEMP')

结果如下图19所示：

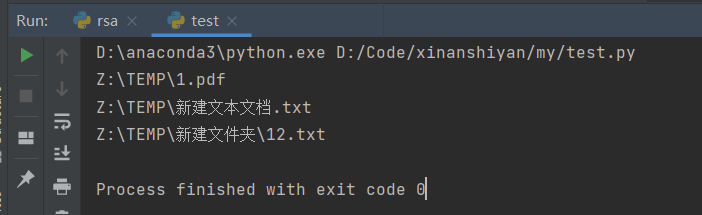


图19验证遍历结果图

**5.2集成测试**

5.2.1测试对U盘进行加密的功能。

在待加密U盘中建立文件B19031614.txt，内容为B19031614，如图20所示。接着对U盘根目录进行加密，如图21所示。加密结束后再次打开B19031614.txt，发现已经变成密文，如图22所示。

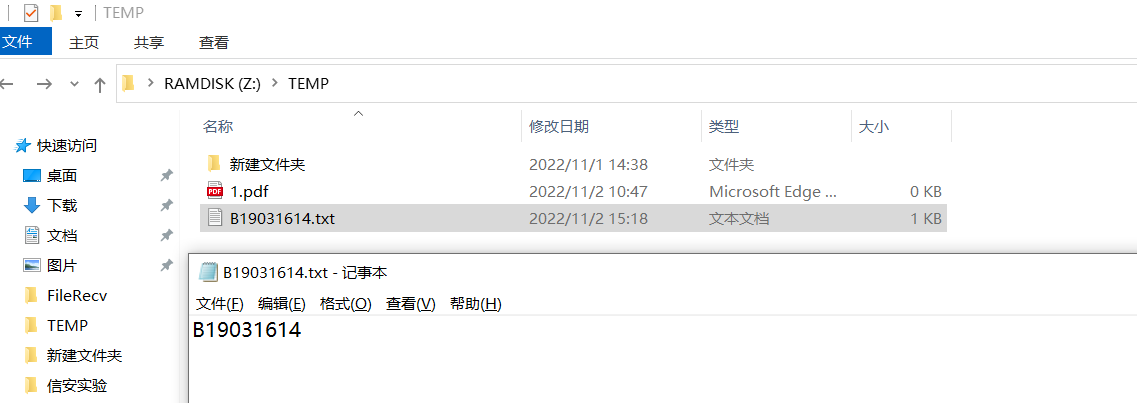


图20明文图

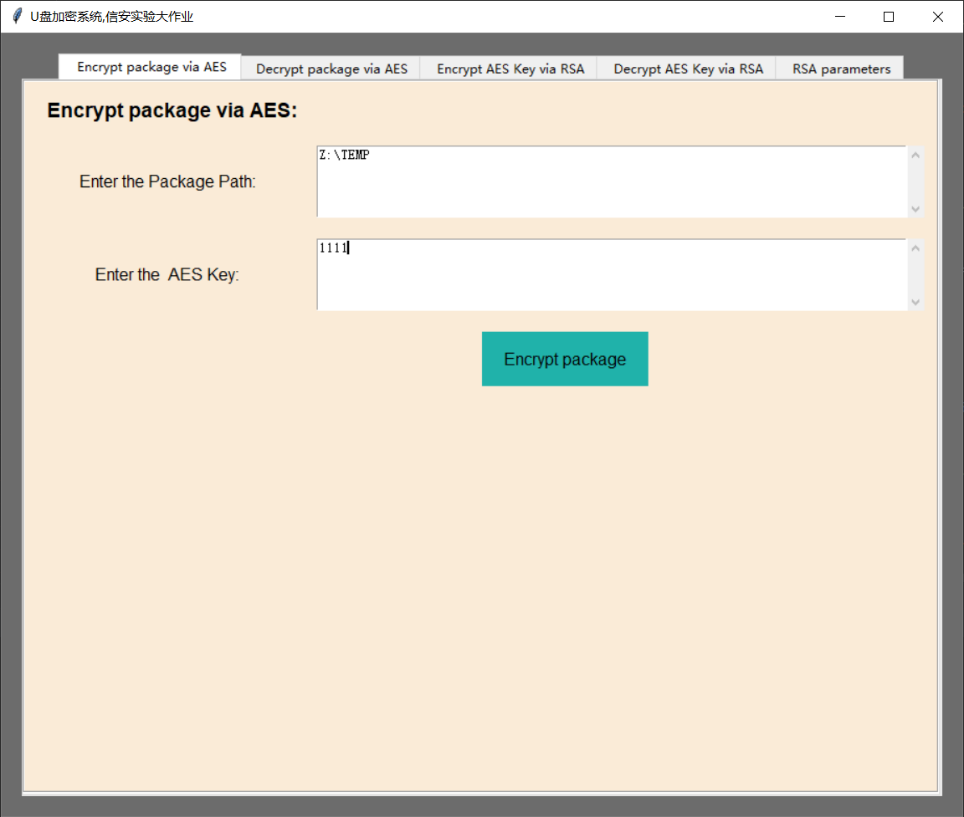


图21加密图

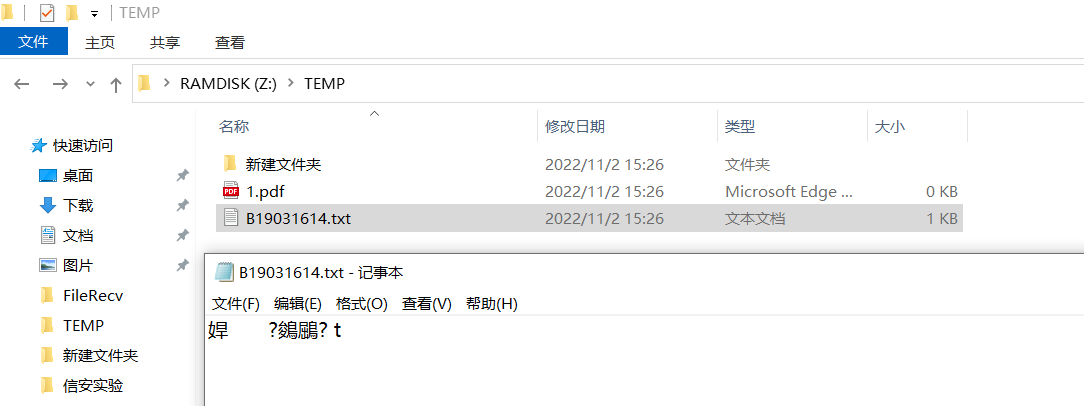


图22密文图

5.2.2测试对U盘进行解密的功能。最终密文被恢复，如图23所示。

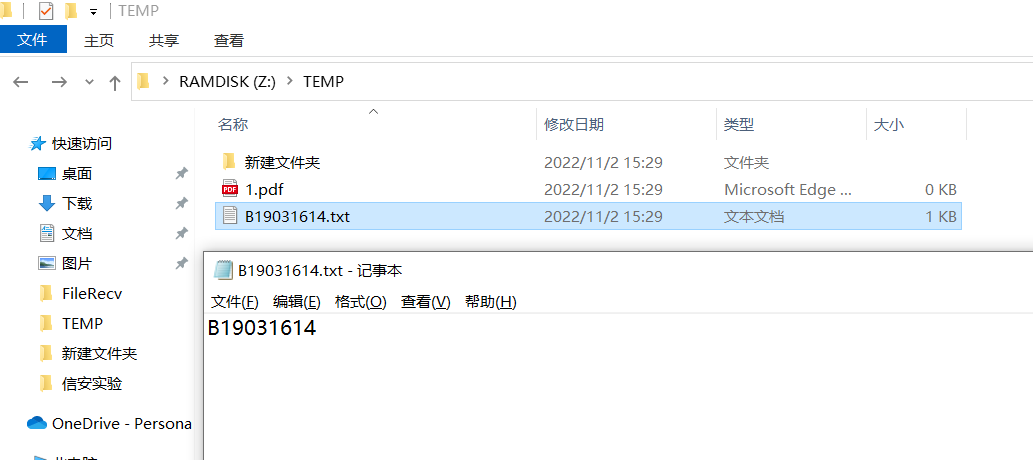


图23恢复明文图

5.2.3 测试对AES公钥进行RSA加密的过程，如图24所示

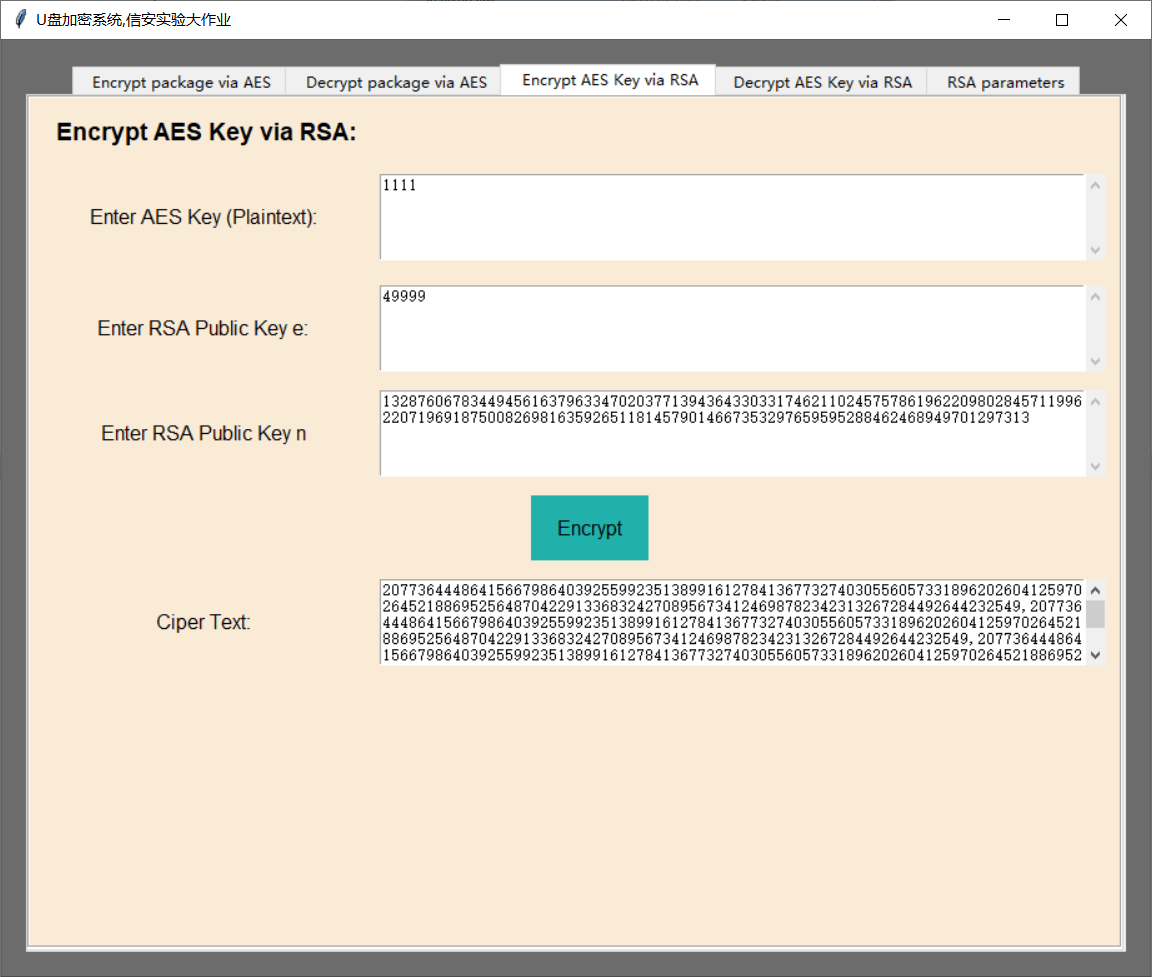


图24 RSA加密图

5.2.4 测试对RSA加密后的AES公钥进行解密的过程，如图25所示

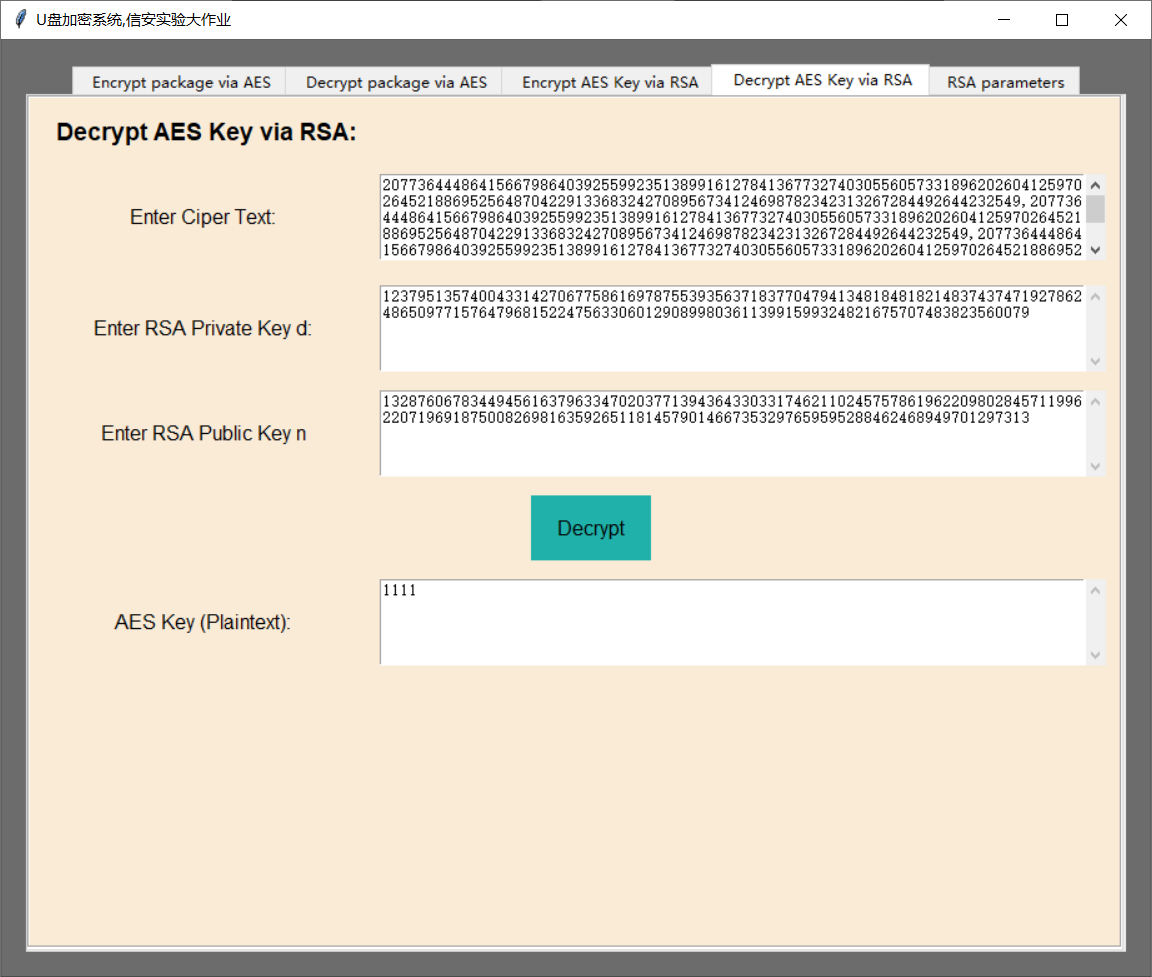


图24 RSA解密图

**六、课题完成过程中遇到的问题及解决方法**

问题1：Python语言中函数内部无法修改超参数（全局变量）

解决方法：使用global关键字，即可在函数内部访问并修改全局变量，如图26所示：

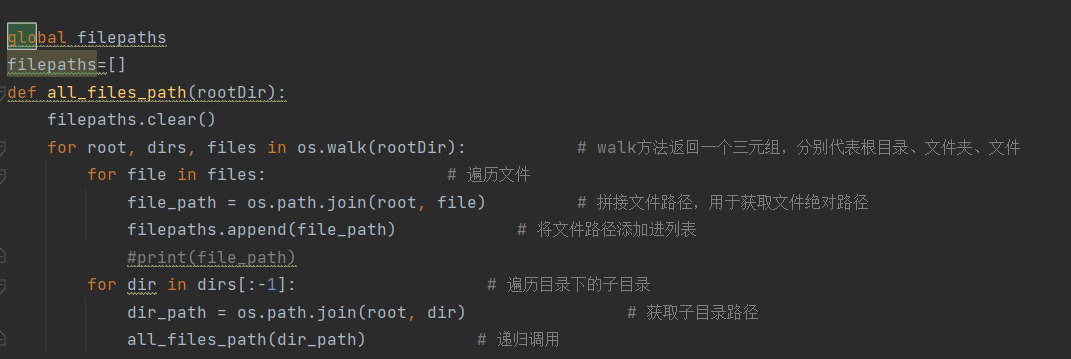


图26 global关键字

问题2：Python报错 UnicodeDecodeError: 'utf-8' codec can't decode byte 0xce in position 52: invalid continuation byte

解决方法：通过查阅资料，发现是Unicode解码错误：“UTF-8”编解码器无法解码位置2中的字节0xBC:无效的起始字节。解决办法是直接用’rb’二进制模式读取文件。

问题3：软件功能需求随着理解的深入不断增多，开发起来复杂

解决方法：采用增量过程模型。首先实现核心产品功能，也就是对指定路径的所有文件进行加密；后续再扩展功能，比如加上通过RSA算法加密AES对称密钥的功能。最后再加上自动生成RSA公钥和私钥的功能。在具体编码时尽量做到高内聚，低耦合的思想。

问题4：关于 RSA算法中明文长度的问题

解决方法：通过查阅资料，发现RSA算法本身要求加密内容也就是明文m必须0<m<n，也就是说内容这个大整数不能超过n，否则就出错。生成密文的长度和明文长度无关， 不管明文长度是多少，RSA 生成的密文长度总是固定的。

问题5：找不到绘制用例图，序列图的免费软件

解决方法：使用在线平台

https://online.visual-paradigm.com/cn/diagrams/features/sequence-diagram-software/

**七、总结**

通过这次实验，我们学习了AES加密的基本原理，并动手开发了一个简单的图形化界面程序，用于对U盘中的全部文件或部分文件进行加密。同时，我们还用RSA公钥加密算法对AES算法中的对称密钥进行加密，以达到保护对称密钥的目的。

AES算法是分组加密算法，这里我们加密的对象是文件。首先将一个文件读取成字节流，再按16字节为一个分组进行划分，最后对每个分组分别调用AES算法加密。RSA算法是公钥加密算法，这里我们加密的对象是字符串（AES算法中的对称密钥），将每个字符转化成其ascill码（int型）以用RSA算法分别加密。

对称加密是最快速、最简单的一种加密方式，加密（encryption）与解密（decryption）用的是同样的密钥（secret key）。对称加密有很多种算法，由于它效率很高，所以被广泛使用在很多加密协议的核心当中。对称加密的一大缺点是密钥的管理与分配，换句话说，如何把密钥发送到需要解密你的消息的人的手里是一个问题。在发送密钥的过程中，密钥有很大的风险会被黑客们拦截。现实中通常的做法是将对称加密的密钥进行非对称加密，然后传送给需要它的人。

公钥加密算法运算速度慢，由于进行的都是大数计算，使得RSA最快的情况也比AES慢上好几倍，无论是软件还是硬件实现，速度一直是RSA的缺陷，所以一般只用于少量数据的加密。RSA的速度是对应同样安全级别的对称[密码算法](https://baike.baidu.com/item/%E5%AF%86%E7%A0%81%E7%AE%97%E6%B3%95)的1/1000左右。

综上，一般使用对称算法来加密数据，然后用RSA来加密对称密钥，然后将用RSA加密的对称密钥和用对称算法加密的消息发送出去。

在实验过程中我们也遇到了很多问题，所幸当下互联网上的资料非常丰富，通过自学我们能解决绝大部分。在写开发程序和写实验报告的过程中，我们也复习并实践了软件工程课程中所学知识。比如需求建模，高内聚低耦合的设计思想以及单元测试集成测试等概念。总而言之，通过此课题我们的各方面能力都得到了提升，衷心感谢《专业课程设计》这门课程给予我们这样一个实践学习的机会。