密码学实验三、 RSA 算法实验报告

班级 572221 学号 LK123425 姓名 黄睿扬

实验目的	1. 掌握 RSA 算法中的密钥产生、加密和解密； 2. 利用 C、C++、Java 实现 RSA 算法。
实验要求	1. 提交实验报告； 2. 提交实验代码。
实验内容	1. 实现 RSA 算法中的密钥产生； 2. 实现 RSA 算法中的加密； 3. 实现 RSA 算法中的解密； 4. 展示实验结果。
实验环境	系统：Windows 11（64 位） 处理器：AMD Ryzen9 7945HX（16 核） IDE：Visual Studio 2022 语言：C++、C 语言 环境：MinGW
实验步骤	一、代码部分 1. RSA 算法概述： RSA 算法是一种非对称加密算法，主要通过一对密钥（公钥和私钥）来实现加密和解密。公钥可公开，私钥必须保密。我们可以通过公钥加密、私钥解密来实现数据的安全加密传输，以及私钥加密、公钥解密的方式来完成对信息的数字签名，达到身份认证的目的。而 RSA 算法的安全性则基于大数分解的困难性。 在本实验中，我建立了 RSA 类，并把 RSA 加密算法需要用到的一些函数，如拓展欧几里得算法（exgcd）、素数判断（isprime）、密钥分发（distributekey）以及加解密函数等封装为 RSA 类的成员函数，以方便其调用。 2. 密钥分发（Key Distribution） 本实验中使用 RSA 类中的 distributekey()函数来生成公钥对和私钥对。该方法首先生成两个大的随机素数 p 和 q。计算 n = p * q 和 φ(n) = (p-1) * (q-1)，其中 φ 是欧拉函数。选择一个小于 φ(n)且与 φ(n)互质的整数 e 作为公钥，利用扩展欧几里得算法计算得到私钥 d。

在实验中，我还考虑到了以下几个方面：

（1）数据类型与取值范围

在 RSA 算法中，素数 p 和 q 的选择至关重要，通常取为较大的整数，因为大素数能够使得私钥难以通过因数分解被破解。。在本实验中，我选取了 long long int 类型作为 p、q 以及明文、密文等数据的数据类型，此数据类型可以存储非常大的数值，适合用来模拟 RSA 加密所需的大素数，并保证这些较大的素数可以自由进行乘法、取模等运算。

同时，在生成公钥和私钥时，由于拓展欧几里得算法会出现 d 为负数的情况，这里通过多次的重新生成的方式，最终得到随机性较好且 e 和 d 均为正数的公、私钥对。

（2）更好的随机性

考虑到通常使用的 time 类型随机数种子的随机性较差，不能够保障密钥生成的安全型，因此经过多方考虑，我在实验中使用随机性更好的 mt19937 生成器。mt19937 是一种基于梅森旋转算法的伪随机数生成器，它具有高度均匀的分布特性，能够提供非常接近均匀分布的随机数，这对于生成素数尤为重要，因为大素数的分布应尽可能随机且不规律。同时，该随机数还具有长周期和高维度均匀性，其周期长度为 2^{19937}，远远超过安全性保障的需求，确保在长序列的随机数生成中不会出现周期性重复。此外相较于其他随机数生成算法，mt19937 在大多数现代计算系统上运行效率高，可以快速生成大量随机数，适合在密钥生成中使用。

3. 加密过程（Encryption）

该过程使用 RSA 类的成员函数 rsa_encryption()来进行加密。该方法首先提示用户输入密钥对（既可以是公钥对也可以是私钥对）和明文，在这个过程中，为了追求实现任意有限长度的明文输入，我采用 string 作为明文输入的类型，然后通过动态内存分配的方式，根据输入的字符串长度，开辟对应大小的内存空间。然后，分段对每个明文字符进行加密，计算 $c = m^e \bmod n$，其中 m 是明文字符的数值。并输出加密后的密文，同时输出密文串的长度（length）。

4. 解密过程（Decryption）

该过程使用 RSA 类中的成员函数 rsa_decryption()来进行解

密。为了方便对应数组的建立和新的动态内存空间的开辟，这里会首先提示用户输入密文串的长度，而前面的加密过程则考虑到了这一点，因此在前面加密过程中会输出对应的密文串长度。根据密文字符串长度建立数组、开辟动态空间后，输入密文，然后用户再输入密钥对（既可以是公钥对也可以是私钥对），对每个密文字符进行解密，计算 $m = c^d \bmod n$，其中 c 是密文字符的数值，然后输出解密后的明文。

5. 用户交互（User Interaction）

为了增强实验的互动性，提高用户的使用体验，我在文件 RSA_Machine 中的 main 函数里加入了简单的命令行交互界面，允许用户选择、密钥分发、加密、解密或退出程序。其中，密钥的分发（选取）是自动的，可以帮助用户快速生成公钥对和私钥对，避免了用户自行选取大素数并进行分解、取模等一系列复杂运算的不必要过程。而通过屏幕提示，用户可以方便地输入明文/密文和密钥，程序则根据用户的选择调用相应的加密或解密函数，并显示操作结果。这不仅使得 RSA 算法的实现更加实用，也为贴心地为用户提供了一个简便的方式来理解 RSA 加密和解密的过程。

二、实验部分

通过实际输入不同的数据进行加密和解密操作，验证 RSA 算法的正确性和效能。

首先，在用户交互提示界面，我们通过选择“1”（自动密钥分发），可以得到一系列的私钥对以及对应的公钥对。

然后，选取其中一组，公钥(21101,10674803)和私钥(2682101, 10674803)，作为我们实验测试用到的密钥对。

接着，我们进行两项测试，分别是“公钥加密、私钥解密”和“私钥加密、公钥解密”。

1. 公钥加密，私钥解密

（1） 选择加密（Encryption），并选择同时包含中文、英文（大小写）、数字、符号的“IloveSEU-57 系!”作为输入的明文数字串；输入公钥(21101,10674803)作为加密密钥，得到加密后的密文串，以及密文串的长度。

（2） 选择解密（Decryption），并先后将上一步生成的密文串长度

和内容输入，再输入私钥(2682101,10674803)作为解密密钥，得到解密后的明文，并与原明文比较，验证大小写、数字、字符、长度是否一致，是否产生乱码。

2. 私钥加密，公钥解密

（1） 选择加密（Encryption），并选择同时包含中文、英文（大小写）、数字、符号的“This’sme-LK123425 黄睿扬!”作为输入的明文数字串；输入私钥(2682101,10674803)作为加密密钥，得到加密后的密文串，以及密文串的长度。

（2） 选择解密（Decryption），并先后将上一步生成的密文串长度和内容输入，再输入公钥(21101,10674803)作为解密密钥，得到解密后的明文，并与原明文比较，验证大小写、数字、字符、长度是否一致，是否产生乱码，能否达到数字签名和身份认证的效果。

最后，选取几组不同的密钥对，多次重复上述过程。并测试在输入错误的密钥时，是否得到错误的明文。

三、实验分析

通过本次 RSA 算法的实验，我们得以深入探讨了 RSA 算法的各个环节，并验证了其在现实应用中的可行性和安全性。实验过程中，我注意到以下几个重要的分析点：

1. 密钥大小的影响：

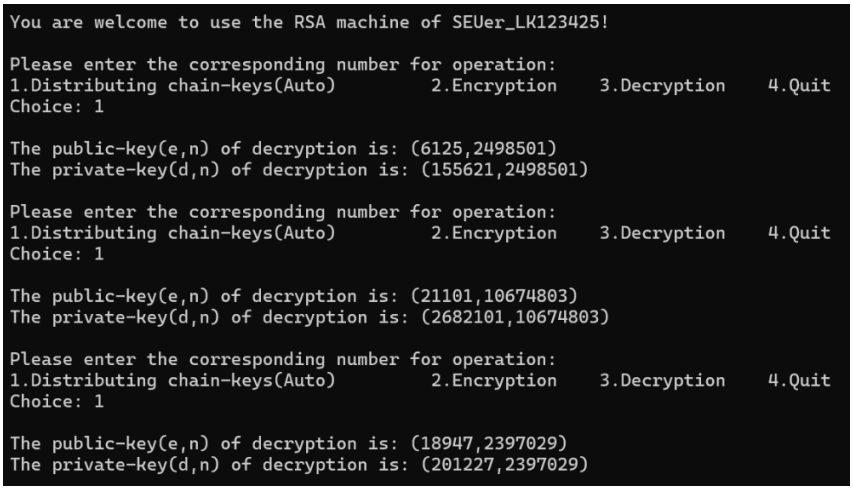
密钥的大小直接影响到加密的安全性。实验中发现，当调节素数表中素数的取值范围，使用较大的质数生成密钥时，破解难度显著增加。这是因为大数分解的复杂性随着数的大小成指数级增长。因此，选择合适的质数大小是保证 RSA 安全性的关键。

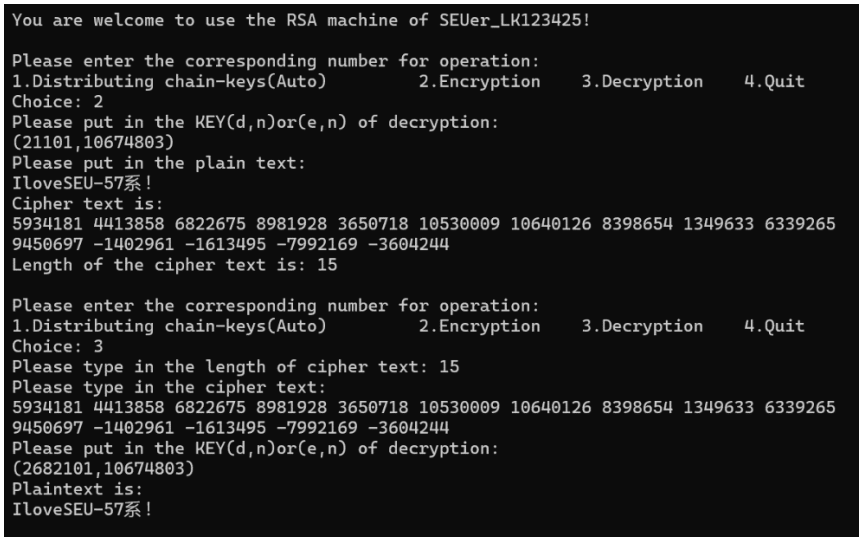
2. 算法的效率：

RSA 算法的计算强度较高，特别是在处理大量数据时。实验中，我测试了不同大小的数据块对加密和解密时间的影响。结果显示，数据块越大，所需时间越长。这一发现提示我们在实际应用中需要平衡安全性和效率。

3. 算法的强度：

通过对加密数据的各种尝试解密（不使用正确的私钥）来测试算法的强度，未能成功破解任何加密数据，证实了 RSA 算法的高安全性。此外，算法对各种类型的输入数据都能正确处理，显示出良好的适应性和稳定性。实验结论

<table>
<tr><td></td><td>四、实验结论
RSA 算法作为一种广泛使用的非对称加密技术，在本实验中展示了其在数据安全和密钥管理方面的优势。通过实际编码和测试，不仅加深了对算法工作原理的理解，也验证了其在处理各种数据类型时的有效性和高安全性。实验结果清楚地表明，适当选择密钥大小和精心设计的算法实现是确保信息安全的关键因素。此外，实验还突出了 RSA 算法在现实应用中需要考虑的效率问题，为将来在更广泛的应用场景中部署提供了重要的实践经验和理论支持。
通过这次实验，我不仅加深了对 RSA 算法的理解，也获得了宝贵的实践经验，为未来的研究和应用奠定了坚实的基础。</td></tr>
<tr><td>实验结果</td><td>1. 生成几组不同的公、私钥对：
 

2. 公钥加密，私钥解密：
 
</td></tr>
</table>

3. 私钥加密，公钥解密：

```
You are welcome to use the RSA machine of SEUer_LK123425!

Please enter the corresponding number for operation:
1.Distributing chain-keys(Auto)         2.Encryption    3.Decryption    4.Quit
Choice: 2
Please put in the KEY(d,n)or(e,n) of decryption:
(2682101,10674803)
Please put in the plain text:
This'sme-LK123425黄睿扬!
Cipher text is:
4652412 946064 6094577 7593587 -3310875 -3864343 7593587 6638885 9249571 7091145 3933325 6191724 4910091 5159612
4524877 5083750 5159612 8225595 -898235 -3832387 -2014582 -7025957 -1905394 -7104060 -7025957 -3310875
Length of the cipher text is: 26

Please enter the corresponding number for operation:
1.Distributing chain-keys(Auto)         2.Encryption    3.Decryption    4.Quit
Choice: 3
Please type in the length of cipher text: 26
Please type in the cipher text:
4652412 946064 6094577 7593587 -3310875 -3864343 7593587 6638885 9249571 7091145 3933325 6191724 4910091 5159612
4524877 5083750 5159612 8225595 -898235 -3832387 -2014582 -7025957 -1905394 -7104060 -7025957 -3310875
Please put in the KEY(d,n)or(e,n) of decryption:
(21101,10674803)
Plaintext is:
This'sme-LK123425黄睿扬!
```

4. 错误输入时，不能得到正确的明文：

```
You are welcome to use the RSA machine of SEUer_LK123425!

Please enter the corresponding number for operation:
1.Distributing chain-keys(Auto)         2.Encryption    3.Decryption    4.Quit
Choice: 1

The public-key(e,n) of decryption is: (3313,6133759)
The private-key(d,n) of decryption is: (3015377,6133759)

Please enter the corresponding number for operation:
1.Distributing chain-keys(Auto)         2.Encryption    3.Decryption    4.Quit
Choice: 2
Please put in the KEY(d,n)or(e,n) of decryption:
(3313,6133759)
Please put in the plain text:
我爱密码学
Cipher text is:
-6035617 -4522544 -4263596 -3899382 -1740482 -2166807 -766154 -6014397 -3359387 -2054413
Length of the cipher text is: 10

Please enter the corresponding number for operation:
1.Distributing chain-keys(Auto)         2.Encryption    3.Decryption    4.Quit
Choice: 3
Please type in the length of cipher text: 10
Please type in the cipher text:
-6035617 -4522544 -4263596 -3899382 -1740482 -2166807 -766154 -6014397 -3359387 -2054413
Please put in the KEY(d,n)or(e,n) of decryption:
(3015376,6133759)
Plaintext is:
鲧鄾獃寘
```

综上所述，该实验能够辅助（大素数生成）随机密钥的分配，并且支持对（理论上）任意长度的字符串的加解密，同时能够实现“公钥加密、私钥解密”和“私钥加密、公钥解密”的功能，并且在错误输入密钥时，不能够正确得到明文。