《应用密码学》实验报告

课程：应用密码学　　实验名称：AES 基本变换

姓名：杨佳倪　　实验日期：2024.4.15

学号：2022132006　　实验报告日期：2024.4.15

班级：信安实验 221

教师评语： 签名： 日期：	成绩：

一、实验名称

RSA 算法的实现

二、实验环境（详细说明运行的系统、平台及代码等）

1．VC

三、实验目的

（1）加深对 RSA 算法的理解；

（2）加深对平方乘算法和模重复平方法的理解；

（3）加深对模块化设计的理解，提高编程实践能力

四、实验内容、步骤及结果

1．实验内容

（1）完成 RSA 算法加密和解密功能；

（2）按照欧几里得扩展算法求，计算 RSA 私钥；

（3）按照平方乘算法和模重复平方法，分别计算 am mod n，完成 RSA 的加密和 解密。

2．实验步骤

主要函数实现方法介绍

1、生成 RSA 私钥的原理如下：

选择两个大素数 p 和 q，并计算它们的乘积 n=p*q。 计算 n 的欧拉函数 ϕ(n)=(p-1)*(q-1)。 选择一个整数 e，使得 1<e< ϕ(n)，且 e 与 ϕ(n)互素。 计算 e 关于 ϕ(n)的模反元素 d，即满足 ed≡1(mod ϕ(n))的整数 d。 公钥为(n, e)，私钥为(n, d)，具体算法：

```
int calculate_private_key_06(int e, int phi_n) {
    int d = 2;
    while ((d * e) % phi_n != 1) {
        d++;
    }
    return d;
}
```

2、模重复平方法原理

算法原理

模重复平方算法是用来快速计算$b^n \mod m$的一个算法。
考虑直接计算$b^n \mod m$，需要$n-1$次乘法，也就是递归计算

$$b^n \equiv (b^{n-1} \mod m) \cdot b \mod m.$$

不过，当n很大的时候，计算会非常耗时。
现在，考虑将n的二进制表示$n = n_0 + n_1 2 + n_2 2^2 + \cdots + n_{k-1} 2^{k-1}$，其中$n_0, n_1, \cdots, n_{k-1}$为0或者1.
则:

$$\begin{aligned} b^n &\equiv b^{n_0 + n_1 2 + n_2 2^2 + \cdots + n_{k-1} 2^{k-1}} \mod m \\ &\equiv b^{n_0} \cdot (b^2)^{n_1} \cdot ((b^2)^2)^{n_2} \cdots (b^{2^{k-1}})^{n_{k-1}} \mod m \end{aligned}$$

通过观察，我们可以知道，我们需要b的偶次幂。设$b_0 = b$的话，下一个偶次幂$b_1 = b_0^2 \mod m$，以此类推，$b_2 = b_1^2 \mod m, \cdots, b_{k-1} = b_{k-2}^2 \mod m$，然后代入上面的公式，有:

$$b^n \equiv b_0^{n_0} \cdot b_1^{n_1} \cdot b_2^{n_2} \cdots b_{k-1}^{n_{k-1}} \mod m.$$

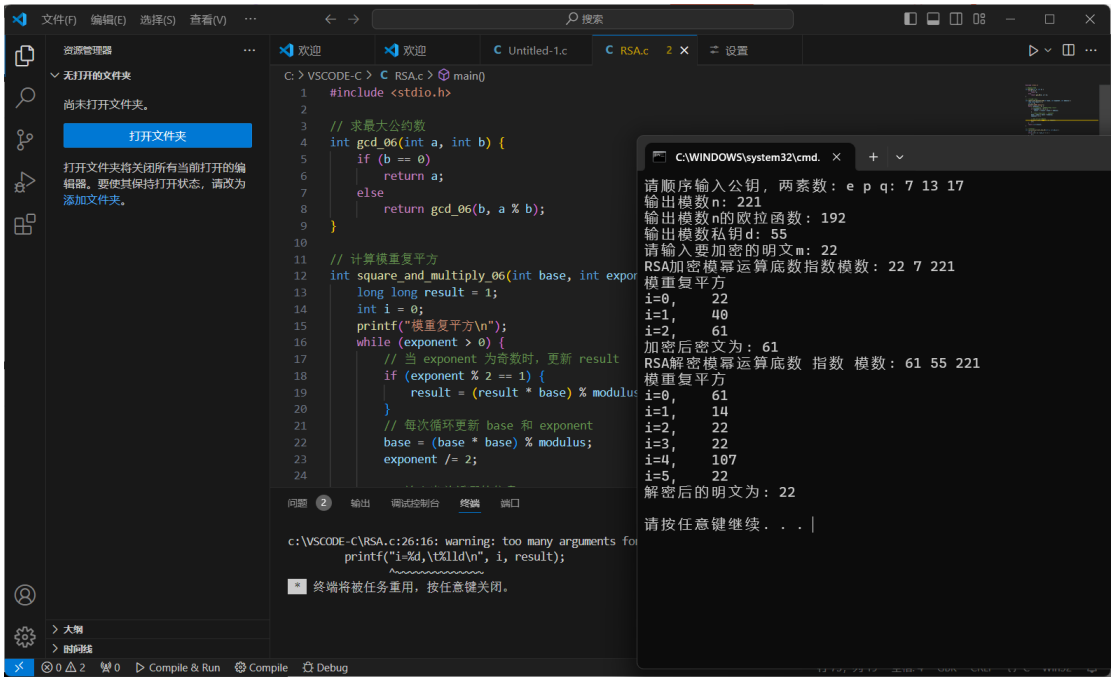
又由于$n_0, n_1, \cdots, n_{k-1}$为0或者1,所以，我们只需要通过不断平方计算出$b_i$，令初始结果为1，然后，如果$n_i$为1则乘上$b_i \mod m$，否则，不管，继续计算$b_{i+1}$.
可以看出，通过上面的计算，我们只需要最多$2\lceil \log_2 n \rceil$次乘法，和一次n的分解。

代码：

```
int square_and_multiply_06(int base, int exponent, int modulus) {
    long long result = 1;
    int i = 0;
    printf("模重复平方\n");
    while (exponent > 0) {
        if (exponent % 2 == 1) {
```

```
            result = (result * base) % modulus;
        }
        base = (base * base) % modulus;
        exponent /= 2;
        printf("i=%d,\t%lld\n", i, result);
        i++;
    }
    return (int)result;
}
}
```

3.实验结果



五、实验中的问题及心得

对于 RSA 算法，在编写程序过程中，核心部分的程序是课程所给示例代码，课后自己也重新扩展完善敲了一遍代码，对 RSA 算法有了更深的了解和掌握，也掌握了 RSA 算法的规则。

附件：程序代码

```c
#include <stdio.h>

// 求最大公约数
int gcd_06(int a, int b) {
    if (b == 0)
        return a;
    else
        return gcd_06(b, a % b);
}
```

```c
// 计算模重复平方
int square_and_multiply_06(int base, int exponent, int modulus) {
    long long result = 1;
    int i = 0;
    printf("模重复平方\n");
    while (exponent > 0) {
        // 当 exponent 为奇数时，更新 result
        if (exponent % 2 == 1) {
            result = (result * base) % modulus;
        }
        // 每次循环更新 base 和 exponent
        base = (base * base) % modulus;
        exponent /= 2;
```

```c
        // 输出当前循环的信息
        printf("i=%d,\t%lld\n", i, result);
        i++;
    }
    return (int)result;
}
```

```c
// 计算私钥 d 值
int calculate_private_key_06(int e, int phi_n) {
    int d = 2;
    while ((d * e) % phi_n != 1) {
        d++;
    }
    return d;
}
```

```c
int main() {
    int e, p, q, n, phi_n, d, m, c;
```

```
    // 读入数据公钥 e 和两素数 p、q 以及待加密消息 m
    printf("请顺序输入公钥，两素数: e p q: ");
    scanf("%d %d %d", &e, &p, &q);

    // 计算 n 值
    n = p * q;

    // 计算 n 的欧拉函数
    phi_n = (p - 1) * (q - 1);

    // 计算私钥 d 值
    d = calculate_private_key_06(e, phi_n);

    // 输出模数 n 和 n 的欧拉函数
    printf("输出模数 n: %d\n", n);
    printf("输出模数 n 的欧拉函数: %d\n", phi_n);

    // 输出模数私钥 d
    printf("输出模数私钥 d: %d\n", d);

    // 输入要加密的明文 m
    printf("请输入要加密的明文 m: ");
    scanf("%d", &m);

    // 使用公钥加密得到密文 c
    printf("RSA 加密模幂运算底数指数模数: %d %d %d\n", m, e, n);
    c = square_and_multiply_06(m, e, n);
    printf("加密后密文为: %d\n", c);

    // 使用私钥解密得到明文 m
    printf("RSA 解密模幂运算底数 指数 模数: %d %d %d\n", c, d, n);
    m = square_and_multiply_06(c, d, n);
    printf("解密后的明文为: %d\n", m);

    return 0;
}
```