RSA实验报告

程序流程

本节我们最重要的就是重载大数，在重载后，程序将会很容易实现

**重载大数类bigInt**

我们设计如下类bigint，其含有一个大小为1025的bool数组，以及代表正负的bool型变量flag

**构造函数**

无参构造函数：全置为0

参数为String的构造函数对每个字符进行判断，每个字符对应0000~1111中的一个，进行遍历赋值

参数为int的构造函数，采用除二取余，进行赋值

**重载运算符**

我们对+、-、\*、/、%、>、<、==进行重载

例如+，我们首先判断符号，对于不同情况进行处理（如a+b,b是负数，则将b符号位取反，return a-b。后续符号处理方法类似，不再进行说明），对于两个都为正的情况，我们用一个temp 存储进位信息，然后循环对每一位进行加法，同时存储进位信息用于下一位加法。

-与+类似

\* 用一个bigint类存储每次相乘的结果result，例如a\*b，循环检测b的每一位是否为1，同时对a进行左移1位，若b的该位为1，则将a加到result中，最后返回结果

/ 与乘法类似，a/b，循环对b左移1位，在循环时，若a>=b，就让a=a-b,并将result的那一位置1，当a<b时循环停止

% 计算a - (a / b) \* b即可

比较运算符号 我们只需要相减然后判断是否全为0即可。

**关键函数**

随机函数 random(int n)生成一个位数为n的数，将最后一位置1，其他位随机生成

模乘函数 modmul（a，b，c）循环 每次循环先计算乘法结果然后取模再进行下一轮

模幂函数

bigint expmod(bigint e, bigint n)

{

bigint c("1");

int i = max - 1;

while (!e[i]) {

i--;

}

int j;

for (j = i; j >= 0; j--) {

c = modMul(c, c, n);

if (e[j]) {

c = modMul(c, (\*this), n);

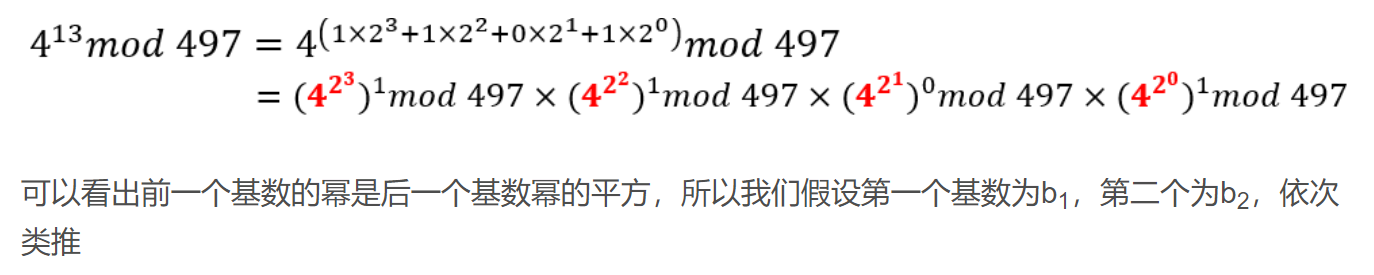
}

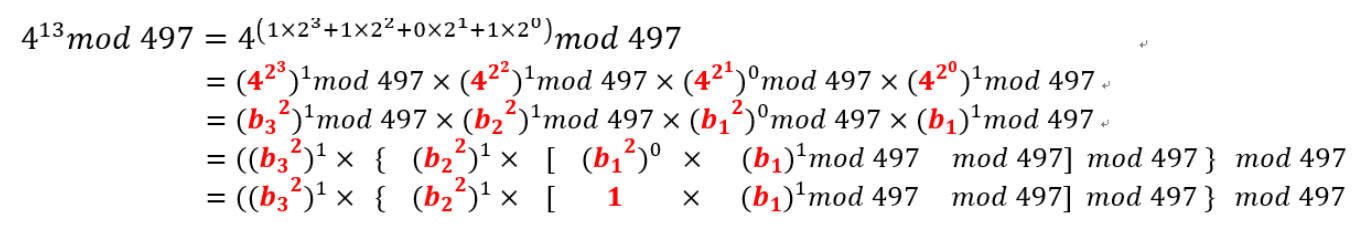
}

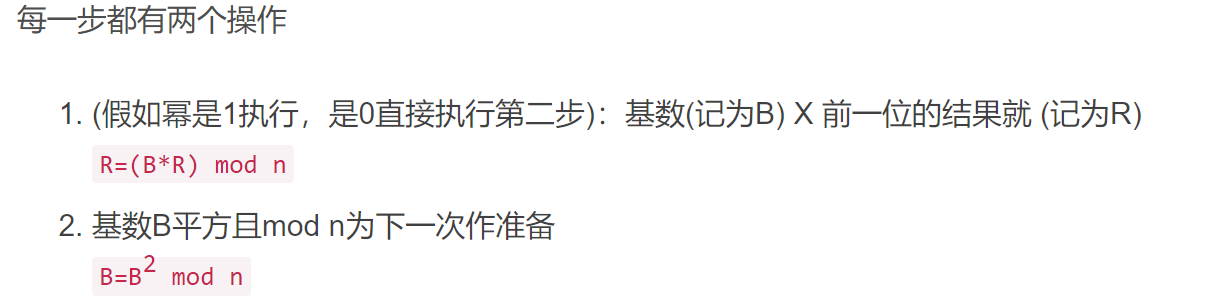
return c;

}

我们采用快速幂进行运算，其逻辑如下







求逆元函数bigint inv(bigint x)

我们采用扩展欧几里得算法求即可

输出函数print（）每四位截取进行输出，对四位进行判断0000~1111

**素数生成器**

在重载大数类后，我们的程序就很简单了

随机数生成器

void random(int n)

{

bigint temp;

srand((int)time(0));

for (int i = 0; i < n; i++)

{

int x = rand() % 2;

if (x == 1)

num[i] = true;

else

num[i] = false;

for (int i = max - 1; i >= n; i--)

{

num[i] = false;

}

num[n - 1] = true;

num[0] = true;

}

判断函数

bool isPrime(bigint a)

{

bigint temp = a;

const bigint ZERO(0);

for (int i = 0; i < 303; i++)

{

bigint p(pri[i]);

bigint d(1);

bigint c = (a % p);

if (c == ZERO)

return false;

}

return true;

}

将2000以内的303个素数进行取余运算，若出现余数为0的情况，则不是素数

程序流程图

生成一个随机数

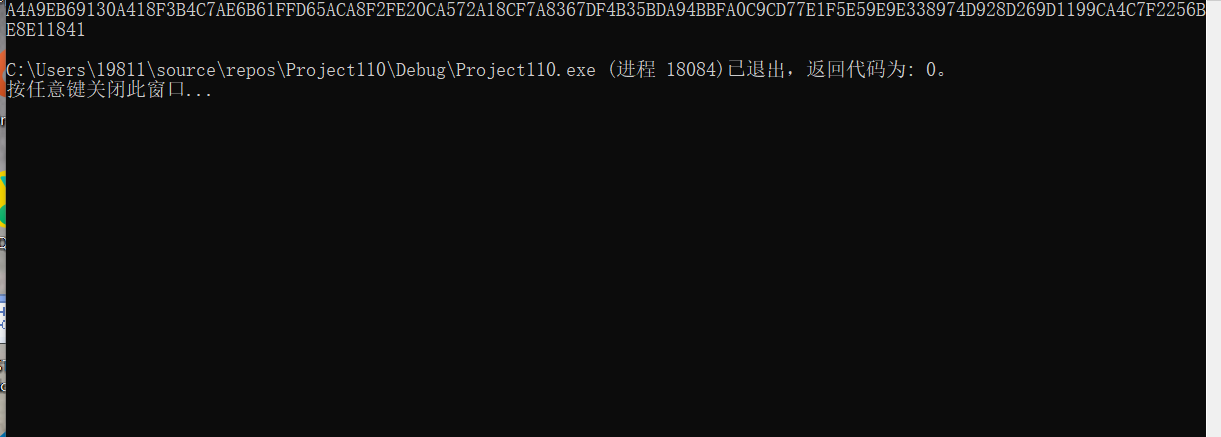
While循环：

调用isprime判断是否是素数，如果是素数返回true，否则返回false

如果是素数，打印，程序退出，

不是素数，则将该数加二

结果



RSA流程图

由于我们已经在大数类里提前预备好了各种函数，所以我们直接调用即可

首先我们获取 p、q两个数，然后计算（p-1）\*（q-1），d 等数据

然后根据用户输入来决定是加密还是解密

然后根据选择使用expmod()的参数得到结果

由于我们产生的素数只是试除了2000以内的素数，可能生成的并不是素数，所以我们这里p、q采用rsa工具生成的素数以验证程序正确性

E就使用工具里默认的10001

解密则输入密文 c

调用decode(c)

计算 m=c.expmod(e,n)

得到明文

加密则输入明文m

调用encode（m）

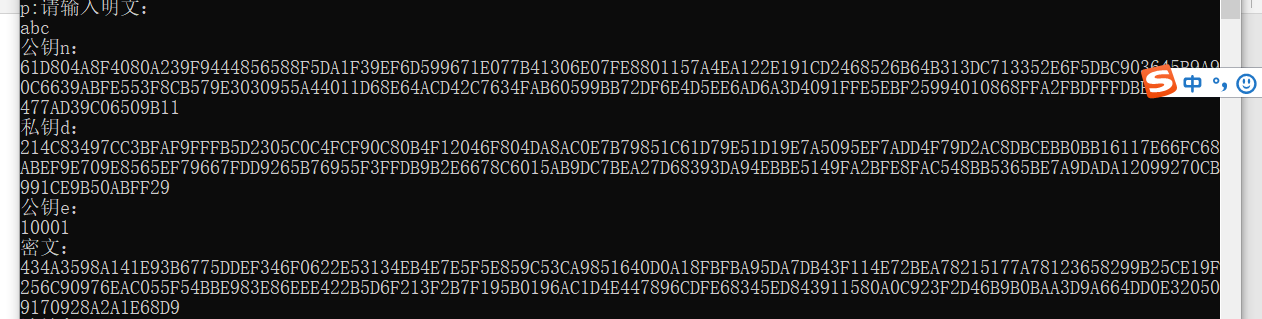
计算c=m.expmod(e,n)

得到密文

得到p、q，根据p、q计算

（p-1）\*（q-1），然后调用inv求乘法逆元计算出d

加密结果



解密结果

