第 2 次编程练习报告

姓名：武桐西　　学号：2112515　　班级：信安一班

一、编程练习 1——平方-乘算法

➢ 源码部分：

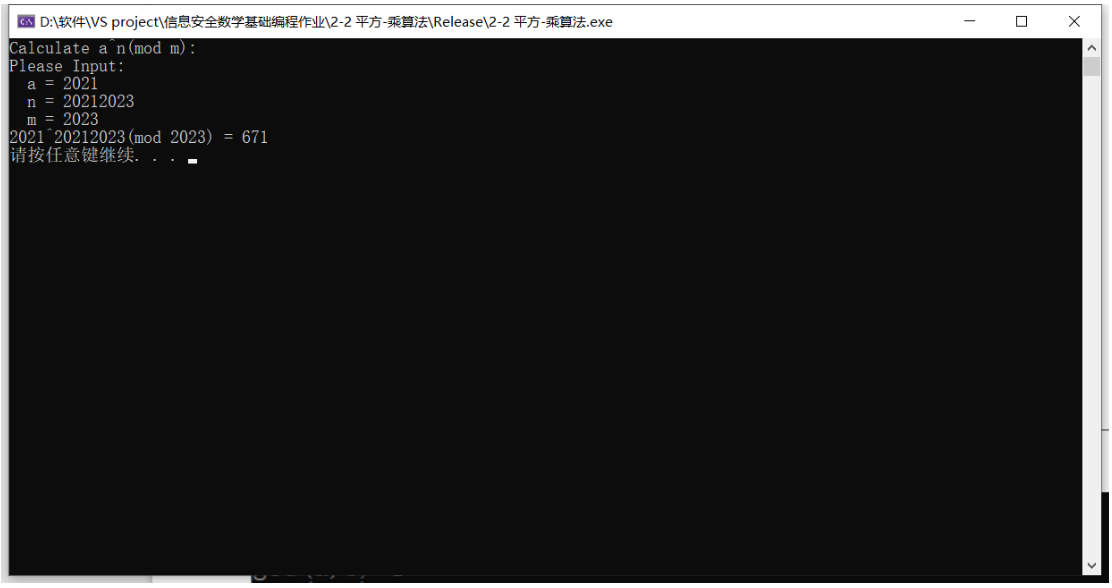
```
#include<iostream>
#include<math.h>
using namespace std;
//平方-乘算法

int FMM(int a, int n, int m) {
    //return a^n(mod m)
    int N = int(log2(1.0 * n)) + 1;//用来计算n转为二进制时的位数
    bool* A = new bool[N];//保存n的二进制
    for (int i = N - 1; i >= 0; i--) {
        A[i] = bool(n % 2);//n转为二进制
        n /= 2;
    }
    int c = 1;//初始化c
    for (int i = 0; i < N; i++) {//从高位开始
        c *= c;
        c %= m;
        if (A[i])
            c = (c * a) % m;
    }
    return c;
}

int main() {
    int a, n, m;
    cout << "Calculate a^n(mod m):\n";
    cout << "Please Input:\n";
    cout << "  a = ";
    cin >> a;
    cout << "  n = ";
    cin >> n;
    cout << "  m = ";
    cin >> m;
```

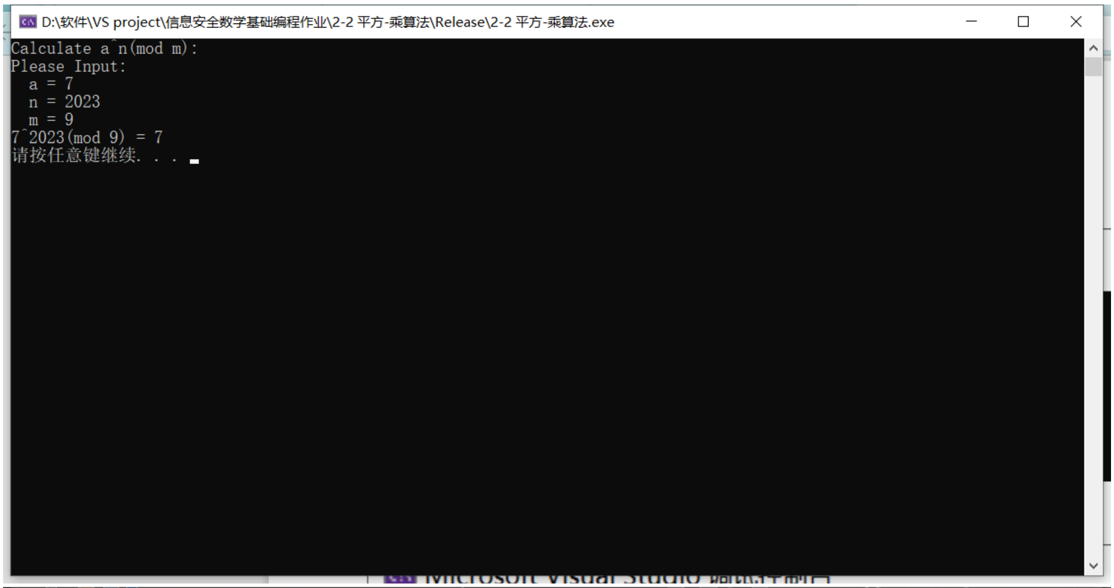
```
    cout << a << "^" << n << "(mod " << m << ") = ";
    cout << FMM(a, n, m) << endl;
    system("pause");//暂停窗口
    return 0;
}
```

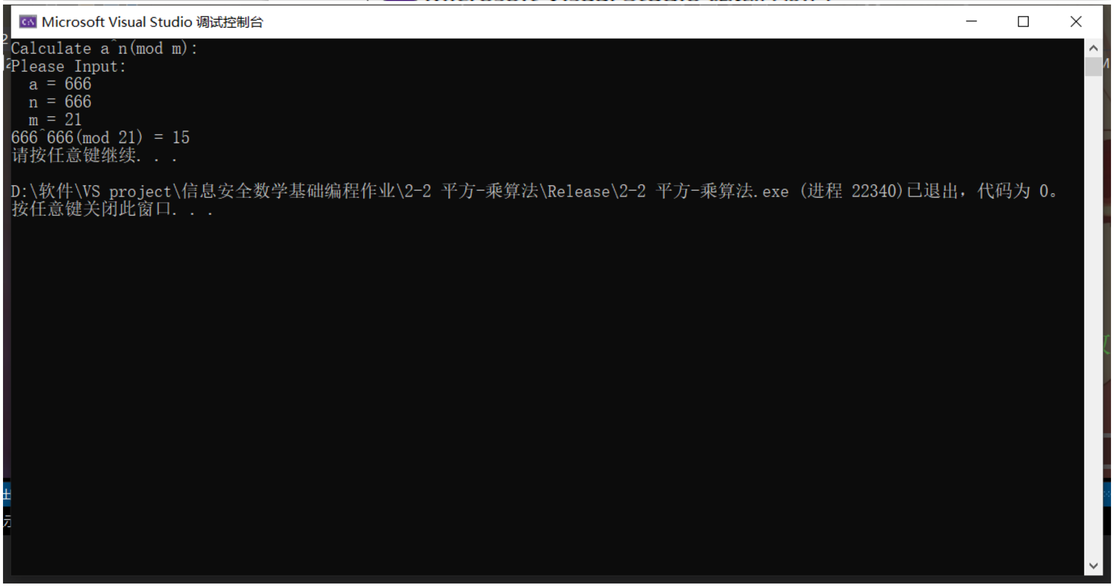
➢ 说明部分：

利用教材中描述的平方-乘算法实现高次模幂运算，其中需要将较高的幂次n转换为二进制表示，在本程序中，利用除2取余法将n转为二进制，并保存在一个 bool 型的数组中（数组的大小在 new 之前通过以2为底的对数$\log_2()$来求得）。

➢ 运行示例：







二、编程练习 2——扩展的欧几里得算法

➢ 源码部分：

```
#include<iostream>
#include<vector>
using namespace std;
//扩展欧几里得算法

void Euclid(int a, int b) {
    vector<int> r;//余数序列
    r.push_back(a > b ? a : b);//a, b中的大者
    r.push_back(a < b ? a : b);//a, b中的小者
    vector<int> q;//商序列
```

```
q.push_back(-1);//q[0]中的值无效
vector<int> s;
s.push_back(1);
s.push_back(0);
vector<int> t;
t.push_back(0);
t.push_back(1);

int x = 0;//索引
while (r[x] % r[x + 1]) {//余数非零，则循环
    r.push_back(r[x] % r[x + 1]);
    q.push_back(r[x] / r[x + 1]);
    s.push_back(s[x] - s[x + 1] * q[x + 1]);
    t.push_back(t[x] - t[x + 1] * q[x + 1]);
    x++;
}

int l = r.size() - 1;//序列的末尾元素下标
cout << "gcd(a, b) = " << r[l] << endl;
cout << "lcm(a, b) = " << (a * b) / r[l] << endl;
if (r[l] == 1) {
    //可用扩展欧几里得算法求逆元(乘法逆元存在)
    if (a > b) {//根据a, b的大小讨论
        //转为最小正缩系中
        if (s[l] < 0)
            s[l] = b + s[l];
        if (t[l] < 0)
            t[l] = a + t[l];
        cout << "a^(-1) = " << s[l] << " (mod " << b << ")\n";
        cout << "b^(-1) = " << t[l] << " (mod " << a << ")\n";
    }
    else {
        //转为最小正缩系中
        if (s[l] < 0)
            s[l] = a + s[l];
        if (t[l] < 0)
            t[l] = b + t[l];
        cout << "a^(-1) = " << t[l] << " (mod " << b << ")\n";
        cout << "b^(-1) = " << s[l] << " (mod " << a << ")\n";
    }
}
else {
    cout << "gcd(a, b) != 1\n";
    cout << "无法通过扩展欧几里得算法求逆元! (即: 乘法逆元不存在)\n";
```

```
    }
}

int main() {
    cout << "Note: the two integers you input must be positive!\n";
    int a, b;
    cout << "a = ";
    cin >> a;
    cout << "b = ";
    cin >> b;
    Euclid(a, b);
    system("pause");//暂停界面
    return 0;
}
```

- **说明部分：**

利用教材中描述的扩展的欧几里得算法求两个正整数a,b的最大公因子、最小公倍数、相互模的乘法逆元。

分别用四个数组r,q,s,t保存余数序列、商序列、s_n序列、t_n序列，依据扩展的欧几里得算法求解即可。最终得到的最后一个非零余数即为$\gcd(a,b)$，利用$lcm(a,b)=\frac{ab}{\gcd(a,b)}$即可求出$a,b$的最小公倍数。若$\gcd(a,b)=1$，则可根据$s_n$和$t_n$得到$a^{-1}(\bmod b)$和$b^{-1}(mod\ a)$。

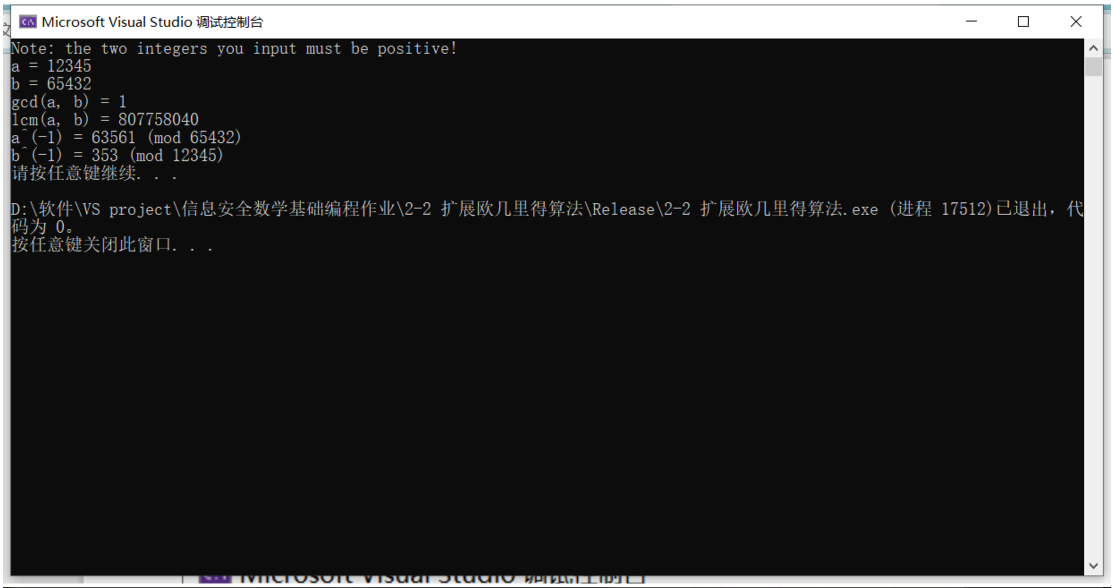
注意：

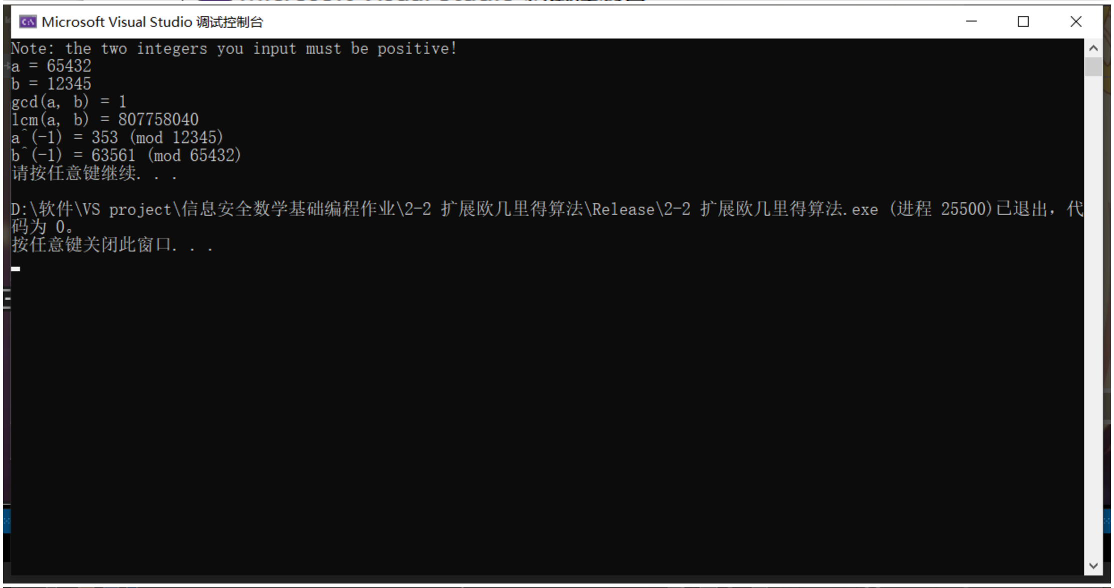
①. $r_0=\max\{a,b\}$，$r_1=\min\{a,b\}$。

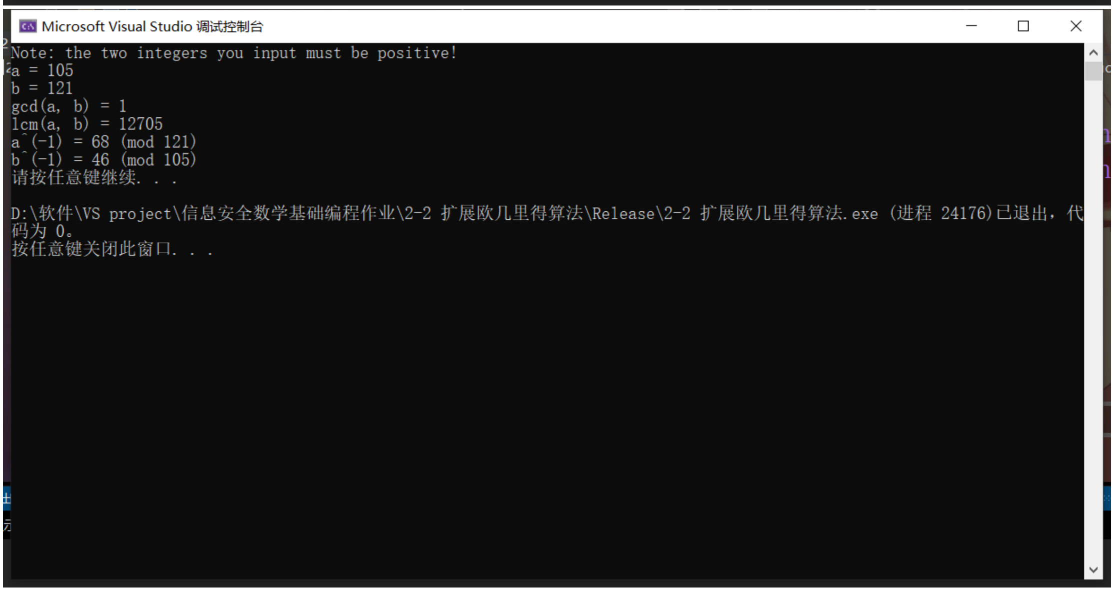
②. 若$\gcd(a,b)\neq 1$，则无法通过扩展的欧几里得算法求乘法逆元，即$a^{-1}(\bmod b)$和$b^{-1}(mod\ a)$不存在。当$\gcd(a,b)=1$时，需要根据a,b的大小关系来进一步得出$a^{-1}(\bmod b)$和$b^{-1}(mod\ a)$。

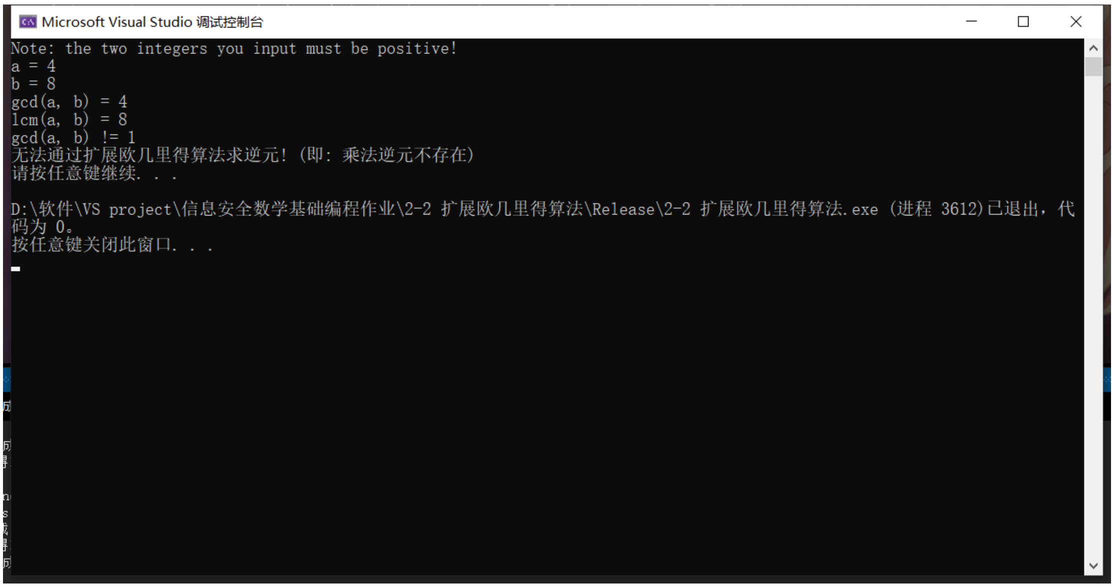
③. 当$\gcd(a,b)=1$时，可以通过扩展的欧几里得算法求相互模的乘法逆元，此时若结果为负数，则可通过当前负数+模数m的方法将其转换到模m的最小正缩系中。

➢ 运行示例：


Microsoft Visual Studio 调试控制台
Note: the two integers you input must be positive!
a = 12345
b = 65432
gcd(a, b) = 1
lcm(a, b) = 807758040
a^(-1) = 63561 (mod 65432)
b^(-1) = 353 (mod 12345)
请按任意键继续. . .
D:\软件\VS project\信息安全数学基础编程作业\2-2 扩展欧几里得算法\Release\2-2 扩展欧几里得算法.exe (进程 17512)已退出，代码为 0。
按任意键关闭此窗口. . .


Microsoft Visual Studio 调试控制台
Note: the two integers you input must be positive!
a = 65432
b = 12345
gcd(a, b) = 1
lcm(a, b) = 807758040
a^(-1) = 353 (mod 12345)
b^(-1) = 63561 (mod 65432)
请按任意键继续. . .
D:\软件\VS project\信息安全数学基础编程作业\2-2 扩展欧几里得算法\Release\2-2 扩展欧几里得算法.exe (进程 25500)已退出，代码为 0。
按任意键关闭此窗口. . .


Microsoft Visual Studio 调试控制台
Note: the two integers you input must be positive!
a = 105
b = 121
gcd(a, b) = 1
lcm(a, b) = 12705
a^(-1) = 68 (mod 121)
b^(-1) = 46 (mod 105)
请按任意键继续. . .
D:\软件\VS project\信息安全数学基础编程作业\2-2 扩展欧几里得算法\Release\2-2 扩展欧几里得算法.exe (进程 24176)已退出，代码为 0。
按任意键关闭此窗口. . .


Microsoft Visual Studio 调试控制台
Note: the two integers you input must be positive!
a = 4
b = 8
gcd(a, b) = 4
lcm(a, b) = 8
gcd(a, b) != 1
无法通过扩展欧几里得算法求逆元！(即：乘法逆元不存在)
请按任意键继续. . .
D:\软件\VS project\信息安全数学基础编程作业\2-2 扩展欧几里得算法\Release\2-2 扩展欧几里得算法.exe (进程 3612)已退出，代码为 0。
按任意键关闭此窗口. . .