

第 2 次编程练习报告

姓名：陆皓喆 学号：2211044 班级：信息安全

一、编程练习 1——平方乘算法

➤ 源码部分：

```
#include<iostream>
using namespace std;
//平方-乘算法
int pingfangcheng(int a, int n, int m) {
    int result = 1;
    while (n > 0) {
        if (n & 1) {
            result *= a;
            result %= m;
        }
        a *= a;
        a %= m;
        n >>= 1; //利用位运算来实现算法
    }
    return result;
}
int main() {
    int a; int m; int n;
    cout << "Calculate a^n(mod m)..." << endl;
    cout << "Please input:" << endl;
    cout << "    a="; cin >> a;
    cout << "    n="; cin >> n;
    cout << "    m="; cin >> m;
    cout << a << "^" << n << "(mod " << m << ")=" << pingfangcheng(a, n, m) << endl;

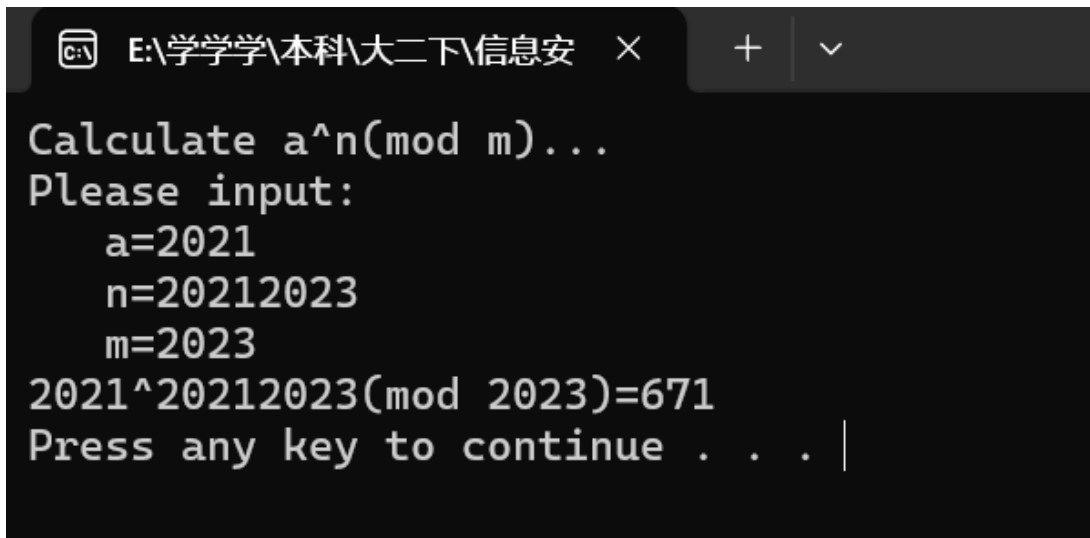
    system("pause");

    return 0;
}
```

➤ 说明部分：

考虑到要将 n 转为二进制，且一位一位计算，我们使用位运算，可以大大简化程序。`pingfangcheng` 函数的功能是，将 n 移位进行计算，每一次都与 1 进行与运算，如果是 1 的话，结果为 1，那么执行 if 语句，将结果先乘 a 再模 m ，否则的话就直接进入 a 自身乘 a 再模 m 。最后，“ $n >= 1$ ”代表将 n 移位，即跳转到下一位二进制数上进行判定与运算。

➤ 运行示例：



```
E:\学学学\本科\大二下\信息安
Calculate a^n(mod m)...
Please input:
a=2021
n=20212023
m=2023
2021^20212023(mod 2023)=671
Press any key to continue . . . |
```

二、编程练习 2——扩展欧几里得算法求逆元

➤ 源码部分：

```
#include<iostream>
using namespace std;
void swap(int &a, int &b) { //change the two numbers
    int temp;
    temp = a;
    a = b;
    b = temp;
}
int oujilide(int a, int b, int& temp1, int& temp2) {
    if (a < b) {
        return oujilide(b, a, temp2, temp1);
    }
}
```

```

int a0 = a; int b0 = b; int q = 1;
int s0 = 1; int s1 = 0; int t0 = 0; int t1 = 1;
while (a % b != 0) {
    q = a / b;
    a = a % b;
    swap(a, b);
    s0 = s0 - q * s1;
    swap(s0, s1);
    t0 = t0 - q * t1;
    swap(t0, t1);

}
temp1 = s1;
temp2 = t1;
if (temp1 <= 0) {
    temp1 = temp1 + b0;
}
if (temp2 <= 0) {
    temp2 = temp2 + a0;
}
return b;
}
int main() {
    int a; int b; int temp1; int temp2;

    cout << "a=";
    cin >> a;
    cout << "b=";
    cin >> b;
    int gcd = oujilide(a, b, temp1, temp2);
    int lcm = a * b / gcd;
    cout << "gcd(a,b)=" << gcd << endl;
    cout << "lcm(a,b)=" << lcm << endl;
    cout << "a^(-1)=" << temp1 << "(mod " << b << ")" << endl;
    cout << "b^(-1)=" << temp2 << "(mod " << a << ")" << endl;
    system("pause");
    return 0;
}

```

➤ 说明部分：

本部分还是采用了上次作业的 gcd 与 lcm 的计算方法，利用书本上提供的公式，我们求出两个值前面的系数即可，我们只考虑前者比后者大的情况，如果前面比后面小的话，就递归调用 oujilide 函数，实现功能。对于求解出来的 temp1 和 temp2，可能会出现不在范围内，比如说是负的，这时候我们就需要将其调整成正的，加上 b0 和 a0 进行调整即可。

➤ 运行示例：



```
E:\学学学\本科\大二下\信息安 × + v
a=12345
b=65432
gcd(a,b)=1
lcm(a,b)=807758040
a(-1)=63561(mod 65432)
b(-1)=353(mod 12345)
Press any key to continue . . . |
```