# 第3次编程练习报告

姓名：陆皓喆 学号：2211044 班级：信息安全

##### 编程练习1——中国剩余定理

* **源码部分：**

#include<iostream>

using namespace std;

void swap(int &a, int &b) {//swap函数实现交换两个值

int temp;

temp = b;

b = a;

a = temp;

}

int oujilide(int a, int b, int& temp1, int& temp2) {//扩展欧几里得算法求逆元

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 Chinese\_remainder\_theorem(int \*b,int \*m, int n,int &M) {//编写中国剩余定理

int\* Mn = new int[n];

int rst = 0;

M = 1;

for (int i = 0; i < n; i++)M \*= m[i];

for (int i = 0; i < n; i++)Mn[i] = M / m[i];

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

{

int temp, nop;

oujilide(Mn[i], m[i], temp, nop);

rst += temp \* Mn[i] \* b[i];

}

delete[]Mn;

rst %= M;//在计算完毕之后，需要取模来获取最后的答案

return rst;

}

int main() {

int n, M;

cout << "n=";

cin >> n;

int\* b = new int[n];

int\* m = new int[n];

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

{

cout << " b\_" << i << "=";

cin >> b[i];

}

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

{

cout << " m\_" << i << "=";

cin >> m[i];

}

int rst = Chinese\_remainder\_theorem(b, m, n, M);

cout << "x≡" << rst << " (mod " << M << ")";

delete[]b;

delete[]m;

return 0;

system(“pause”);

}

* **说明部分：**

中国剩余定理的实现，需要其他算法的支持，如欧几里得求逆元算法，这个算法在上次编程中已经实现过了，所以这里就不再提及。

我们主要分析一下Chinese\_remainder\_theorem函数的实现方式。首先，我们先构造一个大小为n的数组，将M赋值为所有数相乘，然后再分别计算出Mn[i]，即在中国剩余定理的乘数那一项。在计算出Mn[i]后，我们使用oujilide函数来求解出逆元，然后利用rst += temp \* Mn[i] \* b[i]这一句，实现叠加到rst中去。在经过n个数的计算后，我们把各项值都相加起来，最后模上M，就是最后的答案rst。

* **运行示例：**

