# 第三次编程练习报告

姓名：周钰宸 学号：2111408 班级：信安一班

##### **编程练习1——编程实现中国剩余定理**

* **源码部分：**

#include<iostream>

using namespace std;

int ExtendedEuclideanAlgorithm(int a,int b,int &x,int &y) {//扩展的欧几里得算法求逆元，用于求解M'

if (!b)

{

x = 1;

y = 0;

return a;

}

int ret = ExtendedEuclideanAlgorithm(b, a % b, x, y);//递归结束返回时候得到的x和y是下一个状态的，由推导的公式决定此时的x和y

int temp = x;

x = y;

y = temp - a / b \* (y);

return ret;

}

void ChineseRemainderTheorem(int b[], int m[], int n) {

int\* M = new int[n];//用来保存各个M

int m0= 1;//m0用来保存所有mi的乘积m

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

m0\*= m[i];

}

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

M[i] = m0 / m[i];

}

int\* M1 = new int[n];//M1用来保存各个M的逆元

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

int x, y;

ExtendedEuclideanAlgorithm(M[i], m[i],x,y);//调用扩展的欧几里得算法求M的逆元M'

M1[i] = x;

}

int result = 0;

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

result += M[i] \* M1[i] \* b[i];

}

cout << "x≡" << result << " (mod " << m0 << ")" << endl;

return;

}

int main() {

int n;

cout << "n=";

cin >> n;

int\* b = new int[n];//用来存放各个余数b

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

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

cin >> b[i];

}

int\* m = new int[n];//用来存放各个模数m

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

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

cin >> m[i];

}

ChineseRemainderTheorem(b, m, n);

return 0;

}

* **说明部分：**

1. 上一章作业中实现扩展欧几里得算法求逆元中，我的代码实现起来过于复杂，且频繁地调用vector.size会耗费大量的时间。因此本次在实现中国剩余定理的过程中正好涉及了求逆元的过程，**便通过一些数学推导，采用了递归的形式，根本上的大大优化了原来的扩展欧几里得算法**。先给出数学推导如下：

(1)ax1+by1≡gcd(a,b);//由课本P6的定理1.2.3可知

(2)bx2+a%by2≡gcd(b,a%b);//同理可得

(3)gcd(a,b)≡gcd(b,a%b); //运用欧几里得算法中的递归实现，第一次递归时候调用的递归函数为gcd(b,a%b,x,y)

(4)ax1+by1≡bx2+a%b\*y2;//在计算机中，可将a%b写为(a-a/b\*b))

(5)ax1+by1≡bx2+ay2-a/b\*by2;

(6)ax1+by1≡ay2+b(x2-a/b)y2; //合并同类项

(7)x1≡y2,y1≡x2-a/b\*y2;//得出结论

根据以上推导可递归地形式写出扩展欧几里得算法求逆元的函数:

int ExtendedEuclideanAlgorithm(int a,int b,int &x,int &y) 返回值为gcd(a,b)。

2.在已经实现了优化版本的扩展欧几里得算法求逆元后，实现中国剩余定理的函数void ChineseRemainderTheorem(int b[], int m[], int n) 。通过三个数组int b[],int m[],int M[],int M1[]**分别保存中国剩余定理中会涉及到的几个运算量，在这个过程中，通过写好的求逆元函数求解Mi的每个对应逆元Mi’**。

* **运行示例：**

