# 第3次编程练习报告

姓名：申宗尚 学号：2213924 班级：信安班

##### 编程练习1——中国剩余定理

* **源码部分：**
* #include <iostream>  
  #include <vector>  
  using namespace std;  
    
  // 扩展欧几里得算法  
  // 返回值为 (x, y, gcd)，满足 ax + by = gcd  
  tuple<long long, long long, long long> extendedEuclidean(long long a, long long b) {  
   if (b == 0)  
   return {1, 0, a};  
   long long x, y, gcd;  
   tie(x, y, gcd) = extendedEuclidean(b, a % b);  
   return {y, x - (a / b) \* y, gcd};  
  }  
    
  // 计算乘法逆元  
  // 如果存在逆元，返回其值，否则返回 -1  
  long long modInverse(long long a, long long m) {  
   long long x, y, gcd;  
   tie(x, y, gcd) = extendedEuclidean(a, m);  
   if (gcd == 1)  
   return (x % m + m) % m;  
   return -1; // 不存在逆元  
  }  
    
  // 中国剩余定理  
  // 输入 b[], m[] 和 n，其中 b[] 为余数，m[] 为模数，n 为方程个数  
  // 返回满足条件的最小非负整数解，如果不存在返回 -1  
  long long chineseRemainderTheorem(int b[], int m[], int n) {  
   // 计算所有模数的乘积  
   long long M = 1;  
   for (int i = 0; i < n; ++i)  
   M \*= m[i];  
    
   long long x = 0;  
   // 利用CRT计算解  
   for (int i = 0; i < n; ++i) {  
   long long Mi = M / m[i];  
   // 计算Mi的逆元  
   long long Mi\_inverse = modInverse(Mi, m[i]);  
   if (Mi\_inverse == -1)  
   return -1; // 不存在逆元，无法使用CRT  
   x += (b[i] \* Mi \* Mi\_inverse) % M;  
   }  
    
   return (x % M + M) % M;  
  }  
    
  int main() {  
   int n;  
   cout<<"n=";  
   cin >> n;  
   int b[n], m[n];  
   int mod=1;  
   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];  
   mod\*=m[i];  
   }  
    
   long long result = chineseRemainderTheorem(b, m, n);  
   if (result == -1)  
   cout << "无解" << endl;  
   else  
   cout << "x≡" << result << " (mod " << mod <<")" <<endl;  
    
   return 0;  
  }
* **说明部分：**

extendedEuclidean 函数使用扩展欧几里得算法用于求解两个整数 a 和 b 的gcd，以及一对整数 x 和 y，使得 ax + by = gcd(a, b)，从而求解逆元。

函数采用递归实现。基本情况是当 b 等于 0 时，返回 (1, 0, a)，

递归调用过程中，更新 x 和 y 的值，直到得到最终结果。

modInverse 函数计算给定整数 a 在模 m 下的乘法逆元。

函数内部调用 extendedEuclidean 函数来计算 a 和 m 的最大公约数及相应的 x 和 y 值。

如果最大公约数等于 1，说明 a 在模 m 下存在逆元，返回逆元值 (x % m + m) % m，否则返回 -1。

chineseRemainderTheorem 函数使用中国剩余定理求解出最终答案

函数接受一个余数数组 b[]，一个模数数组 m[]，以及方程个数 n。

首先计算所有模数的乘积 M和缺少某个模数的Mi，然后对Mi求其逆元，如果某个模数的乘法逆元不存在，则无法使用中国剩余定理，返回 -1。

最后使用中国剩余定理，返回解 x。

* **运行示例：**

