电子科技大学信息与软件工程学院

**实 验 报 告**

学 号 2017221302006

姓 名 周玉川

（实验） 课程名称 信息安全数学基础

理论教师 陈大江

实验教师 陈大江

**电子科技大学教务处制表**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名：周玉川 学号：2017221302006 指导教师：陈大江**

**实验地点：信软楼西304 实验时间：2018.11.22**

**一、实验名称：**模乘法逆元的实现。

**二、实验学时：2。**

**三、实验目的：**

1.掌握模乘法逆元的求法，学习扩展的欧几里得定理，以及欧拉定理。

2.利用所学的编程语言，以及扩展欧几里得欧拉定理实现求模的逆元。

**四、实验原理：**

1.当x =a-1 mod m存在时，由定义a\*x = 1 mod m，很显然存在整数y使得a\*x + m\*y = 1成立，再根据扩展的欧几里得定理，所求的解x即为a的系数。

2.当a，m互素时，x =a-1 mod m存在，由欧拉定理am = 1 mod m，显然 x = a-1 = am-1 mod m，即x = am-1 mod m，利用上一次实验模指数运算可求得x。

**五、实验内容：**

本实验要求学生掌握常用的模乘法逆元的实现方法，并运用高级程序设计语言完成一种求逆算法的程序，加深对求模乘法逆运算的理解。

**六、实验器材（设备、元器件）：**

Pc一台。

**七、实验步骤：**

1. 编写代码。

2. 输入要求逆元的数，以及模的数字。

3. 输出该数模下的逆元。

**八、实验结果与分析（含重要数据结果分析或核心代码流程分析）**

程序代码

|  |
| --- |
| package 同余  import java.util.Scanner;  /\*  \* 求逆元，有两种方法，一个是利用欧拉定理a^m = 1(mod m)所以a^-1 = a^(m-1)(mod m)  \* 另一种方法用扩展的欧几里得（递归）  \*/  public class 求逆元 {  // 重复平方算法  static int repeatedSquareAlgorithm(int a, int k, int m) {  int b = 1;  if (k == 0) {  } else {  int A = a;  if (k % 2 == 1) {  b = a;  }  k = k / 2;  while (k != 0) {  A = (A \* A) % m;  if (k % 2 == 1) {  b = (A \* b) % m;  }  k = k / 2;  }  }  return b;  }  // 重复平方算法结束  // 递归的扩展欧几里得算法  static Demo extendsEuclidenAlgorithm(int a, int q) {  // 当q为0时，到达递归终点，返回最大公因子，以及x =1,y=0  if (q == 0) {  Demo tDemo = new Demo(a, 1, 0);  return tDemo;  } else {  // 得到下一层的Demo  Demo tDemo = extendsEuclidenAlgorithm(q, a % q);  // 推导出父层与子层的gcd，x，y关系  int temp = tDemo.x;  tDemo.x = tDemo.y;  tDemo.y = temp - (int) (a / q) \* tDemo.y;  // 得到该层的Demo 并返回给上一层  return tDemo;  }  }  public static void main(String[] args) {  // TODO Auto-generated method stub  int a, m;  Scanner xScanner = new Scanner(System.in);  System.out.println("输入该数字以及要模的数字，a,m(求解x = a^-1 mod m)");  a = xScanner.nextInt();  m = xScanner.nextInt();  xScanner.close();  // 利用扩展的欧几里得求解  Demo tDemo = extendsEuclidenAlgorithm(a, m);  System.out.println(tDemo + "\t利用扩展欧几里得算法:\n" + a + "模" + m + "的逆元为" + tDemo.x);  // 利用欧拉定理求解x = a^(m-1) mod m  int x = repeatedSquareAlgorithm(a, m - 1, m);  System.out.println("\n利用欧拉定理求解x = " + a + "^" + (m - 1) + "mod" + m + ":\n" + a + "模" + m + "的逆元为" + tDemo.x);  }  }  //里面储存扩展欧几里得系数  class Demo {  int x, y;  int gcd;  // 构造器初始化  public Demo(int gcd, int x, int y) {  this.gcd = gcd;  this.x = x;  this.y = y;  }  // 改变x，y  public void setData(int gcd, int x, int y) {  this.gcd = gcd;  this.x = x;  this.y = y;  }  } |

代码分析

1. 求逆元，有两种方法，一个是利用欧拉定理a^m = 1(mod m)所以a^-1 = a^(m-1)(mod m)，关键代码

|  |
| --- |
| static int repeatedSquareAlgorithm(int a, int k, int m) {  int b = 1;  if (k == 0) {  } else {  int A = a;  if (k % 2 == 1) {  b = a;  }  k = k / 2;  while (k != 0) {  A = (A \* A) % m;  if (k % 2 == 1) {  b = (A \* b) % m;  }  k = k / 2;  }  }  return b;  } |

1. 另一种方法用扩展的欧几里得（递归），关键代码

|  |
| --- |
| static Demo extendsEuclidenAlgorithm(int a, int q) {  // 当q为0时，到达递归终点，返回最大公因子，以及x =1,y=0  if (q == 0) {  Demo tDemo = new Demo(a, 1, 0);  return tDemo;  } else {  // 得到下一层的Demo  Demo tDemo = extendsEuclidenAlgorithm(q, a % q);  // 推导出父层与子层的gcd，x，y关系  int temp = tDemo.x;  tDemo.x = tDemo.y;  tDemo.y = temp - (int) (a / q) \* tDemo.y;  // 得到该层的Demo 并返回给上一层  return tDemo;  }  } |

结果分析

|  |
| --- |
| 输入该数字以及要模的数字，a,m(求解x = a^-1 mod m)  4864 3458  同余.Demo@28d93b30 利用扩展欧几里得算法:  4864模3458的逆元为32  利用欧拉定理求解x = 4864^3457mod3458:  4864模3458的逆元为32 |

结果正确，实验成功。

**九、总结及心得体会：**

1.这次实验在扩展的欧几里得算法的基础上，自己增加了根据欧拉定理求解逆元的方法，还成功求解，很开心。

2.兴趣对学习的帮助是极大的，很大可能事半功倍。

3.代码能力占一定重要性，关键在于理解力。

4.第一次实验用C语言编写，第二，三次实验用java编写，争取下一次用python编写。

**十、对本实验过程及方法、手段的改进建议：**

1.实验内容较少，书本上除了这个重复平常算法，还有Garner算法（中国剩余定理），以及同余方程，勒让德符号等均可以用代码实现，可以适当增加内容。

2.鼓励学生用C语言之外的编程语言完成要求。

3.数学知识尤其是和算术相关的定理，用计算机语言很容易实现，若有兴趣，可以把代码写出来，很有趣，也有成就感，对信安基础的学习也大有帮助。

4.希望老师能一次少布置一点作业，但是分多次布置一章的作业，最好一星期交一次作业，这样不容易遗忘知识。

**报告评分：**

**指导教师签字：**