**电子科技大学信息与软件工程学院**

**实 验 指 导 书**

**（实验）课程名称 信息安全数学基础实验**

**电子科技大学教务处制表**

目 录

[实验一 多精度整数运算的实现 16](#_Toc372579955)

[实验二 模指数运算的实现 18](#_Toc372579956)

[实验三 模乘法逆元的实现 21](#_Toc372579957)

[实验四 有限域上运算的实现 22](#_Toc372579958)

# 实验一 多精度整数运算的实现

**一、实验目的：**

* 1. 掌握一种1024位大数运算原理；
  2. 依据所选择的算法，编程实现该算法；
  3. 提供该算法的使用说明手册，可执行代码、源代码及测试用例（包括测试用例手册、可执行代码和源代码）。

**二、实验内容：**

本实验要求学生掌握常用的大数运算算法的实现方法，并运用高级程序设计语言完成一种大数运算算法的程序，加深对大数运算的理解。

**三、实验步骤：**

1. 熟悉大数表示的数学原理；
2. 进行算法需求分析；
3. 制定算法流程图；
4. 根据算法需求分析和算法流程图编写实现代码；
5. 编译并运行。

**四、实验要求：**

实验之前，理解如下知识点：

大数指的是超过计算机CPU寄存器表达的数，即超过计算机字长的数。大数基本运算主要指的是对大数进行数论运算，如加、减、乘、除。出于效率原因，一般的大数运算主要指对无符号类型的数进行数论计算。

目前主流RSA算法都建立在512位到1024位的大数运算之上。然而大多数的编译器只能支持到64位的整数运算，即运算中整数必须小于等于64位，即：0xFFFFFFFFFFFFFFFF，这远远达不到RSA的需要，于是需要建立专门的大数运算库来解决这一问题。

最简单的办法是将大数当作字符串处理，也就是将大数用10进制字符数组进行表示，然后模拟人们手工进行“竖式计算”的过程编写其加减乘除函数。但是这样做效率很低，因为1024位的大数其10进制数字个数就有数百个，对于任何一种运算，都需要在两个有数百个元素的数组空间上做多重循环，还需要许多额外的空间存放计算的进位退位标志及中间结果。其优点是算法符合人们的日常习惯，易于理解。

另一种方法是将大数当作一个二进制流进行处理，使用各种移位和逻辑操作来进行加减乘除运算，但是这样做代码设计非常复杂，可读性很低，难以理解也难以调试。

本实验要求学生掌握常用的大数运算算法的实现方法，并运用高级程序设计语言完成一种大数运算算法的程序，加深对大数运算的理解。

类及其成员函数、变量接口说明如表1.。

表1.4类及其成员函数、变量接口说明

|  |  |
| --- | --- |
| 类名 | BigNum |
|  | |
| 成员函数 | public int bigPlus (int a[],int b[],int c[]) |
| 名称 | 大数加法运算 |
| 接口 | int bigPlus (int a[],int b[],int c[]) |
| 输入 | int a[],int b[] |
| 输出 | int c[] |
| 功能 | 计算大数a+b |
| 出错 | 返回-1 |
|  | |
| 成员函数 | public int bigSub (int a[],int b[],int c[]) |
| 名称 | 大数减算 |
| 接口 | int bigSub (int a[],int b[],int c[]) |
| 输入 | int a[],int b[] |
| 输出 | int c[] |
| 功能 | 计算a-b |
| 出错 | 返回-1 |
|  | |
| 成员函数 | public int bigMult (int a[],int b[],int c[]) |
| 名称 | 大数乘算 |
| 接口 | int bigMult (int a[],int b[],int c[]) |
| 输入 | int a[] ，int b[] |
| 输出 | int c |
| 功能 | 计算a\*b |
| 出错 | 返回-1 |

# 实验二 模指数运算的实现

**一、实验目的：**

* 1. 熟悉一种模指数运算算法；
  2. 运用高级程序设计语言完成一种模指数运算算法的程序，加深对模指数运算的理解；
  3. 提供该算法的使用说明手册，可执行代码、源代码及测试用例（包括测试用例手册、可执行代码和源代码）。

**二、实验内容：**

* 1. 本实验要求学生掌握常用的模指数算法的实现方法，即**：**求ae mod m 的值。算法采用重复平方乘的方法实现。

**三、实验步骤：**

1. 熟悉模指数运算的数学原理；
2. 进行算法需求分析；
3. 制定算法流程图；
4. 根据算法需求分析和算法流程图编写实现代码；
5. 编译并运行。

**四、实验要求：**

实验前要做好充分准备，理解如下知识点：

1、模的指数运算可以运用重复平方乘算法有效实现，这一运算在很多密码学协议中都有重要用处。这一算法是基于以下结果，设k的二进制表示为，其中，则





**例2.1**利用重复平方乘算法计算。

**解：**将重复平方乘按步骤进行分解，如下表所示

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i |  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ki |  | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| A |  | 5 | 25 | 625 | 681 | 1011 | 369 | 421 | 779 | 947 | 925 |
| b |  | 1 | 1 | 625 | 625 | 67 | 67 | 1059 | 1059 | 1059 | 1013 |

表2.1重复平方乘算法计算

因此

。

2、M-ary乘方算法

通过对重复平方乘算法的推广，M-ary算法是把指数用大于2的基表示，并在重复平方乘步骤4中用乘以a的幂的乘法代替乘以a的乘法。设指数e由对某一确定的基M的表示由e=(en-1en-2,…e0)M给出。下面的算法计算幂**ae mod m**。

求**a**e **mod m** 的M-ary算法如下：

* + 1. 计算并作为一张表格存储**a2 mod m, a3 mod m,… aM-1 mod m**。
    2. 置**p←ae**n-1,且**i←n-2**。
    3. 置**p←p**M **mod m**。
    4. 若**ei≠0**，置**p←p\*aei mod m**。
    5. 置**i←i-1**；若**i≧0**转到步骤③。
    6. 输出**p**。

所需乘法的次数很明显依赖指数e的位数，因此依赖M的选择。所以我们这样确定M：使在步骤③中的乘方最大可能地通过求平方计算，并且使a的方幂的预计算中的乘法次数最少，这样表格的存储空间的代价才可以接受。

# 实验三 模乘法逆元的实现

**一、实验目的：**

* 1. 熟悉一种求乘法逆元算法；
  2. 依据所选择的算法，编程实现该算法；
  3. 提供该算法的使用说明手册，可执行代码、源代码及测试用例（包括测试用例手册、可执行代码和源代码）。

**二、实验内容：**

本实验要求学生掌握常用的求乘法逆元算法的实现方法，并运用高级程序设计语言完成一种求乘法逆元运算算法的程序，加深对求乘法逆元运算的理解。

**三、实验步骤：**

1. 熟悉求乘法逆元的核心算法扩展的欧几里德算法的数学原理；
2. 进行算法需求分析；
3. 制定算法流程图；
4. 根据算法需求分析和算法流程图编写实现代码；
5. 编译并运行。

**四、实验要求：**

实验之前，理解如下知识点：

扩展Euclid算法原理如下：

计算gcd(a,n)和a mod n,a≧0,n>0的乘法逆元的扩展Euclid算法描述如下：

① 置u←1,g←a,v1←0,及v3←n。

② 通过带余除法由g=q\*v3+t3且t3<v3计算q和t3，并置t1←u-q\*v1 mod n,u←u1,g←v3,v1←t1及v3←t3。

③ 若v3=0，输出g为gcd(a,n)，u为a mod n的逆并终止算法；否则，转到步骤②。

模n的步骤t1←u-q\*v1 mod n保证t1,v1和u不变为负数。最后有u∈[1,…,n-1]。

# 实验四 有限域上运算的实现

**一、实验目的**

* 1. 加深对有限域结构的理解，实现常用的有限域上元素的基本运算。；
  2. 编程实现GF（28）域中元素的运算，提供使用说明手册，可执行代码、源代码及测试用例（包括测试用例手册、可执行代码和源代码）。

**二、实验内容**

实现28域上元素的多项式基表示，实现模多项式的乘法运算和求逆运算，从而实现28域上元素乘法运算和逆元运算。更进一步，构造指数对数表，从而通过查表实现28域上元素乘法运算和逆元运算。

**三、实验步骤**

1. 熟悉28域中元素的表示；
2. 熟悉多项式模乘运算和求逆运算；
3. 进行算法需求分析；
4. 制定算法流程图；
5. 学习相关的类及相关成员的建立；
6. 根据算法需求分析和算法流程图编写实现代码；
7. 编译并运行。

**四、实验要求**

实验前具备以下基本知识：

取域上的8次不可约多项式，是的一个根。因此有限域可以表示为的所有次数小于8的多项式集合，即



定义一个由组成的字节可表示为系数为{0，1}的二进制多项式：



还可以将每个字节表示为一个16进制数，即每4比特表示一个16进制数，代表较高位的4比特的符号仍在左边。例如，01101011可表示为6B。同样也可以用0-255这256个十进制整数来表示域中的元素。它们之间的运算为中的运算，其加法定义为二进制多项式的加法，且其系数模2，其乘法定义为多项式的乘积模一个次数为8的不可约多项式。通过计算机实验可以验证元素“02”是域中的一个本原元。

将域中的元素用0-255这256个十进制整数来表示，指数对数表可以通过如下步骤来构建：

（1）将元素‘02’表示成为，依次计算，，将所得结果转变为十进制数，设为，；如下表所示：

（2）建表。第一行为，第二行元素依次为，。由于，约定第2行，第255列元素为0。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 |  | 253 | 254 | 255 |
| 1 | 2 | 4 | 8 |  | 233 | 177 | 0 |

（3）按所建表的第二行元素的大小进行重排列，如下表所示：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 255 | 0 | 1 | 197 |  | 72 | 230 | 104 |
| 0 | 1 | 2 | 3 |  | 253 | 254 | 255 |

（4）将（3）中表的第一行放在（2）中表的第三行，即

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 0 | 1 | 2 | 3 |  | 253 | 254 | 255 |
|  | 1 | 2 | 4 | 8 |  | 233 | 177 | 0 |
|  | 255 | 0 | 1 | 197 |  | 72 | 230 | 104 |

建立上述指数对数表之后，通过查表很容易求出两个元素的乘积。又由于对于均有，所以可通过查表也很容易求出元素的逆元。