**电子科技大学信息与软件工程学院**

**标 准 实 验 报 告**

**（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**

**电 子 科 技 大 学**

**实 验 报 告**

**学生姓名： 陆圣珩 学 号：2017221302009 指导教师： 陈大江**

**实验地点：3教410 实验时间：9.23**

**一、实验室名称： 网络安全实验室**

**二、实验项目名称：多精度数运算的实现**

**三、实验学时：2学时**

**四、实验原理：**

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。

**五、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；

**六、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。

**七、实验步骤：**

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 |

程序代码如下：

public int bigSub (int a[],int b[],int c[]){

int i, j, len, blag;

char \*temp;

int n2[MAX] = {0};

char num1 = str(a);

char num2 = str(b);

int len1 = strlen(num1);

int len2 = strlen(num2);

if(len1 < len2) {

blag = 1;

temp = num1;

num1 = num2;

num2 = temp;

len = len1;

len1 = len2;

len2 = len;

}

else if(len1 ==len2) {

for(i = 0; i < len1; i++){

if(num1[i] == num2[i])

continue;

if(num1[i] > num2[i]) {

blag = 0;

break;

}

else{

blag = 1;

temp = num1;

num1 = num2;

num2 = temp;

break;

}

}

}

len = len1>len2 ? len1 : len2;

for (i = len1-1, j = 0; i >= 0; i--, j++)

c[j] = num1[i] - '0';

for (i = len2-1, j = 0; i >= 0; i--, j++)

n2[j] = num2[i] - '0';

for (i = 0; i <= len; i++)

c[i] = c[i] - n2[i];

if (c[i] < 0){

c[i] += 10;

c[i+1]--;

}

}

for (i = len1-1; i>=0 && c[i] == 0; i--)

;

len = i+1;

if(blag==1) {

c[len] = -1; len++;

}

return len; }

public int bigMult (int a[],int b[],int c[]){

char num1[] = char(a);

char num2[] = char(b);

int i, j, len, len1, len2;

int a[MAX+10] = {0};

int b[MAX+10] = {0};

int c[MAX\*2+10] = {0};

len1 = strlen(num1);

for(j = 0, i = len1-1; i >= 0; i--)

a[j++] = num1[i]-'0';

len2 = strlen(num2);

for(j = 0, i = len2-1; i >= 0; i--)

b[j++] = num2[i]-'0';

for(i = 0; i < len2; i++){

for(j = 0; j < len1; j++){

c[i+j] += b[i] \* a[j];

}

}

for(i=0; i<MAX\*2; i++) {

if(c[i]>=10)

{

c[i+1]+=c[i]/10;

c[i]%=10;

}

}

for(i = MAX\*2;

len = i+1;

for(; i>=0; i--)

c[i]=c[i];

}

**九、实验数据及结果分析：**

实现范例测试界面如图1.1，测试界面代码请参考附录。该示范程序用VC++高级语言实现。使用时依次在a,b后面输入数值，点击加，减，乘计算按钮，计算结果在其后面显示出来。

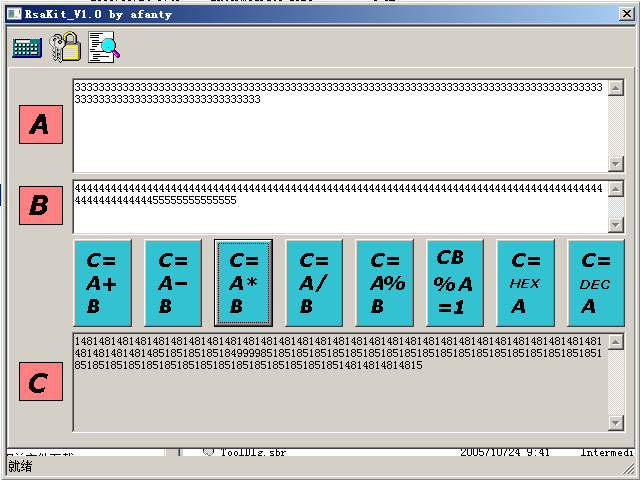


图1.1大数实现范例

图1.1 为两个64bit以上长度的数进行乘法运算所得结果。用软件中的除法进行验证，结果正确。

**十、实验结论：**

采用数组的方法，结合数组操作，成功实现大整数的加、减、乘、除以及其他相关运算。

**十一、总结及心得体会：**

大数运算主要是将大数转换为b进制表示，利用数组进行存储再进行处理。

**十二、对本实验过程及方法、手段的改进建议：**

建议在A,B输入时，显示其长度或二进制表示的位数。

**报告评分：**

**指导教师签字：**