**大整数运算**

**--数据结构大型实验报告**

**作者：**

**Rush Chuh**

****

**完成日期：2014.12.22**

2014

目录

[1 实验内容分析 2](#_Toc406984401)

[1.1 内容及要求 2](#_Toc406984402)

[1.2 相关图示 3](#_Toc406984403)

[1.2.1 类的结构 3](#_Toc406984404)

[1.2.2 类间关系 5](#_Toc406984405)

[1.2.3 程序流程图 6](#_Toc406984406)

[1.2.4 调用关系 7](#_Toc406984407)

[1.3 算法简述 7](#_Toc406984408)

[1.3.1 加法 7](#_Toc406984409)

[1.3.2 减法 8](#_Toc406984410)

[1.3.3 乘法 9](#_Toc406984411)

[1.3.4 除法 10](#_Toc406984412)

[1.3.5 取余运算 11](#_Toc406984413)

[1.3.6 指数运算 11](#_Toc406984414)

[2 实验验证分析 12](#_Toc406984415)

[2.1 输入形式和输入值范围 12](#_Toc406984416)

[2.1.1 输入形式 12](#_Toc406984417)

[2.1.2 输入值范围 12](#_Toc406984418)

[2.2 输出的形式 12](#_Toc406984419)

[2.3 程序功能 12](#_Toc406984420)

[2.4 测试数据 13](#_Toc406984421)

[3 调试分析 13](#_Toc406984422)

[3.1 主要问题及解决方法 13](#_Toc406984423)

[3.2 技术难点分析 14](#_Toc406984424)

[3.3 印象最深的调试错误及修正方法 14](#_Toc406984425)

[4 测试结果 15](#_Toc406984426)

[4.1 普通数据 15](#_Toc406984427)

[4.2 边界数据 18](#_Toc406984428)

[5 附录 21](#_Toc406984429)

[bigInteger.h 21](#_Toc406984430)

[linkedList.h 22](#_Toc406984431)

[interface.h 24](#_Toc406984432)

[bigInteger.cpp 24](#_Toc406984433)

[linkedList.cpp 37](#_Toc406984434)

[interface.cpp 44](#_Toc406984435)

[main.cpp 80](#_Toc406984436)

[6 开发人员 81](#_Toc406984437)

# 1 实验内容分析

## 内容及要求

大整数的运算

**【问题描述】**密码学分为两类密码：对称密码和非对称密码。对称密码主要用于数据的加/解密，而非对称密码则主要用于认证、数字签名等场合。非对称密码在加密和解密时，是把加密的数据当作一个大的正整数来处理，这样就涉及到大整数的加、减、乘、除和指数运算等，同时，还需要对大整数进行输出。请采用相应的数据结构实现大整数的加、减、乘、除和指数运算，以及大整数的输入和输出。

**【基本要求】**

1. 要求采用链表来实现大整数的存储和运算，不允许使用标准模板类的链表类(list)和函数。同时要求可以从键盘输入大整数，也可以文件输入大整数，大整数可以输出至显示器，也可以输出至文件。大整数的存储、运算和显示，可以同时支持二进制和十进制，但至少要支持十进制。大整数输出显示时，必须能清楚地表达出整数的位数。测试时，各种情况都需要测试，并附上测试截图；
2. 要求大整数的长度可以不受限制，即大整数的十进制位数不受限制，可以为十几位的整数，也可以为500多位的整数，甚至更长；
3. 大整数的运算和显示时，只需要考虑正的大整数。如果可能的话，请以秒为单位显示每次大整数运算的时间；
4. 要求采用类的设计思路，不允许出现类以外的函数定义，但允许友元函数。主函数中只能出现类的成员函数的调用，不允许出现对其它函数的调用。
5. 要求采用多文件方式：.h文件存储类的声明，.cpp文件存储类的实现，主函数main存储在另外一个单独的cpp文件中。如果采用类模板，则类的声明和实现都放在.h文件中。
6. 要求源程序中有相应注释；
7. 不强制要求采用类模板，也不要求采用可视化窗口；
8. 要求测试例子要比较详尽，各种极限情况也要考虑到，测试的输出信息要详细易懂，表明各个功能的执行正确；

要求采用Visual C++ 6.0及以上版本进行调试。

## 相关图示

### 类的结构

**节点类**



½Úµã

Êý¾Ý

next

pre

**链表类**



**大整数类**



### 1.2.2 类间关系

****

### 1.2.3 程序流程图

****

### 1.2.4 调用关系

* 加法：

调用了链表类中的无参构造函数和在头部添加元素的函数。

* 减法：

调用了链表类中的无参构造函数、在尾部添加元素的函数和大整数类中的比较大小函数。

* 乘法：

调用了链表类中的无参构造函数、在头部添加元素的函数。

* 除法：

调用了链表类中的无参构造函数、在头部和尾部添加元素的函数和大整数类中的比较大小函数。

* 取余：

因为和除法大同小异，故这里不做赘述，只是修改了返回值。

* 指数：

调用了链表类中的无参构造函数、在头部添加元素的函数和大整数类中的比较大小函数、乘法、减法。

## 算法简述

### 加法

【算法】从两个链表的尾部开始向前遍历，并且逐位相加，遇到进位的情况就先将相加的结果对10取余，之后将rest置为1，进位完成后再将rest置为0，以免干扰后面的运算。

【图示】



### 减法

【算法】和加法是大同小异，不过之前要比较一下两个数的大小，如果是小数减大数则先将一个名为isMinus的布尔型变量置为true，之后再让大数减小数。后面的过程与加法相同我就不再赘述了，唯一不同的就是如果借位了就将rest置为-1之后让上面一位先加10再减下面一位，完成这一位的运算后再将rest还原为0。

【图示】



### 乘法

【算法】这里我用到了一种叫“卷积法”的方法，首先将下面的链表的指针放在表尾，上面的链表从后向前遍历相乘，一轮过后，将下面链表的指针向前推一位，再让上面的链表从后向遍历相乘，最后将乘好的数全部相加放到一个临时的链表里，紧接着最关键的来了，让这个卷积化的链表规范化，就完成了两个数的相乘。

【图示】



### 除法

【算法】首先截取被除数中少于除数1位的链表作为起始的余数，然后让原来指向被除数的指针后跳，再截取一位，之后让这个临时的余数减去除数，一直减到比除数小为止，记录循环次数作为商。

【图示】



### 取余运算

【算法】和除法一样的过程，只要把返回值改变即可，这里不做过多说明了。

【图示】见除法的图示。

### 指数运算

【算法】假设计算a^b：for(int i=次数,tmpResult=1;i>0;i--)

tmpResult=tmpResult\*a;

只要将这个for循环改写成链表形式的即可。

【图示】略。

# 实验验证分析

## 输入形式和输入值范围

### 输入形式

1. 输入第一个数
2. 输入第二个数
3. 输入需要进行取余运算的数的底数
4. 输入需要进行取余运算的数的指数

### 输入值范围

首先是正负数的问题。除了减法可以运算出负数，其他的运算都是不支持负数进行运算的。

其次是效率的问题。理论上来讲只要时间足够充裕，计算机足够快，多少位都是可以进行计算的，但是经过我的测试发现，计算6000位的数大概需要花费7分钟。所以如果超过这个位数很多的运算，需要权衡一下时间的花销能否接受。在后面的测试数据部分将会给出截图。

## 输出的形式

采用了国际通用的三位分节法，并且在结束之后还有“共计多少位”的字样显示，一目了然。在后面的测试数据部分将会给出截图。

## 程序功能

可以进行理论上不限位数的长整数的加、减、乘、除及指数运算，并且支持从文本读入数据和导出计算结果到文本。另外为了响应老师的要求，还在每个运算结束之后可以对从键盘输入的指数进行取模运算以检验数据的正确性。

## 测试数据

* 加法：

999999999+999999999=？

该组测试数据是测试加法中的特殊情况。

* 减法：

00000000099999999-22222222222222=？

该组测试数据是测试减法中负数的情况以及自动忽略前面的0的功能。

* 乘法：

111111111\*111111111=？

该组测试数据是测试乘法的正确性。

* 除法：

9009009/9=？

该组测试数据是测试除法中对0的处理及除法的正确性。

* 指数：

2^4000=？

该组测试数据是测试算法的性能。

# 调试分析

## 主要问题及解决方法

1. 在开始写这个实验的时候对老师要求中提到的要写链表类和大整数类这两个类不是很理解，后来问过老师才知道链表类负责链表的基本操作和存储数据，而大整数类则通过对链表的调用和修改来处理大整数的各种运算。
2. 在编写乘法的时候，怎么修改都是有bug，卡在这里一天的功夫，最终放弃了初始的算法，用了一种新的方法：卷积法。这种方法也就是报告开篇进行图示的方法。它的优点就是简单，不容易计算错误，也可以减少调试程序的成本，但是缺点也比较明显，就是需要计算机更多的运算成本。
3. 在起初编写双向链表的时候也遇到了问题，由于老师布置过相应的作业，内容是单向链表。所以在第一次编写双向链表的时候就遇到了各种问题，经常导致一些本来可以直接实现的语句出现bug，最后是在对运算过程中的调试才发现原来是最基础的链表的功能实现没有考虑周到，下次可以从中吸取教训，每次写好一个函数都对它进行全方位的测试，务必保证写出来的东西是可以用的。

## 技术难点分析

1. 程序的流程我觉得是一个难点。当时在想流程的时候由于运算比较多而且大一的大型实验也没有亲力亲为的缘故，所以也不知道编框图都需要写哪些内容，所以就想到一点编一点，最后参考了下大一的代码中的结构部分之后把这次的框图编完，同时还学会了一个小技巧，可以打印出类似进度条的东西，只需要包含进windows.h的包，调用其中的sleep函数加上for循环跟cout语句即可实现这样的效果。
2. 起初在思考指数的算法时，想到了要将指数变成乘法来运算，但是没想到的是，次数要怎么计算，因为原来用的都是int型的，但是后来突然发现，只需要在开始定义一个值为1的链表就可以了，同时配合链表的减法就可以实现这一运算。
3. 最恶心的要属除法了，我编了一遍又一遍，可总是遇到bug，要不就是用3位数模拟除法编出来后只能用在3位数上，要不就是对0的考虑不周全。直到现在我还是在参考一个同学的代码下编出来的，看来以后还应该多加强编程能力。

## 印象最深的调试错误及修正方法

1. 首当其冲的就是去除前面的0了，开始我编好之后发现调用display函数总是出问题，后来经过短点调试之后才发现原来我把所有的0都删掉了，结果链表成了空链表，后来加了对size的判断才正确。
2. 在乘法中，开始编写了一个程序也是出现了指针为空的bug，原因就是，有一条语句在最初执行算法的时候不能执行，之前在上课时听老师讲到了哨兵一词，突然想到可以用这个方法来解决这个问题。
3. 最后的这个错误应该是印象最深的一个错误了，这个错误的原因是在编写指数运算的时候通过断点调试找到的。从指数运算中找到了减法，再从减法又找到拷贝构造函数。终于在拷贝构造函数里发现了问题：本来等号是可以自动调用拷贝构造函数的，但是在更新链表对象时，却在开始没有对链表进行清空操作，导致了后面的一系列错误，后来跟小组同学讨论了一下这个问题，得知将清空函数写在拷贝构造函数中也不符合规范。拷贝构造函数只是用来在最初创建链表对象的时候调用的，后面对链表的操作理论上来讲都不应该在这里出现，因此重载了等号，实现了这一功能。

# 测试结果

## 4.1 普通数据

* 加法：

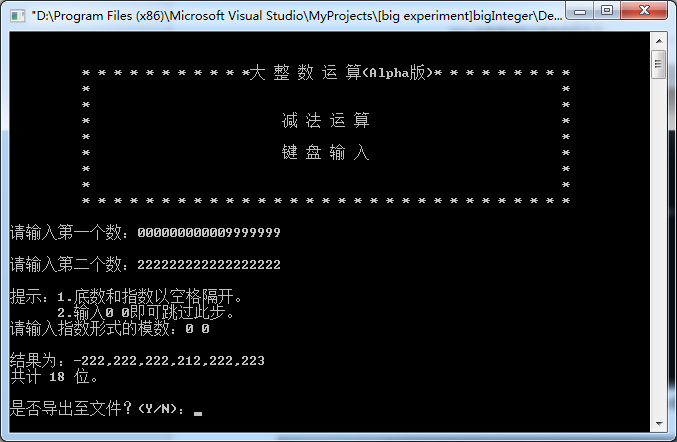


该组测试数据是测试加法中的特殊情况。



该组测试数据是测试从文件输入的大数加法运算。

* 减法：



该组测试数据是测试减法中负数的情况以及自动忽略前面的0的功能。



该组测试数据是测试从文件输入的大数减法。

* 乘法：



该组测试数据是测试乘法的正确性。



该组测试数据是测试从文件输入的大数乘法。

* 除法：



该组测试数据是测试除法中对0的处理及除法的正确性。



该组测试数据是测试从文件输入的大数除法。

* 指数：



该组测试数据是测试算法的性能。



该组测试数据是测试大数的指数运算。

## 4.2 边界数据

* 加法：



* 减法：



* 乘法：



* 除法：



* 指数：



# 附录

## bigInteger.h

//bigInteger.h

#ifndef BIGINTEGER\_HEADER

#define BIGINTEGER\_HEADER

#include<string>

#include"linkedList.h"

using namespace std;

//大整数类：运算

class bigInteger

{

public:

//add

linkedList add(const linkedList &left,const linkedList &right);

//subtract

linkedList subtract(const linkedList &left,const linkedList &right);

//比大小：1是前面大，2是后面大，0是一样大

int isLarge(const linkedList &left,const linkedList &right);

//multiply

linkedList multiply(const linkedList &left,const linkedList &right);

//divide

linkedList divide(const linkedList &left,const linkedList &right);

//mod

linkedList mod(const linkedList &left,const linkedList &right);

//power

linkedList power(const linkedList &left,const linkedList &right);

};

#endif

## linkedList.h

//linkedList.h

#ifndef LINKEDLIST\_HEADER

#define LINKEDLIST\_HEADER

#include<string>

using namespace std;

//节点类

class node

{

public:

int data;

node \*next,\*pre;

node():next(0),pre(0){}

node(const int &item,node \*nextnode=0,node \*prenode=0):data(item),next(nextnode),pre(prenode){}

};

//链表类：存储数据

class linkedList

{

public:

//默认构造函数

linkedList();

//构造函数

linkedList(string &s);

//创建指定长度的链表

void create(int lsize);

//拷贝构造函数

linkedList(const linkedList &l);

//重载=

linkedList& operator=(const linkedList& rhs);

//析构函数：释放动态申请的空间

~linkedList();

//清空链表

void clear();

//输出链表函数

void display();

//导出链表

void savetxt(const string &path);

//在头部添加一个元素

void push\_front(const int &item);

//在尾部添加一个元素

void push\_back(const int &item);

friend class bigInteger;

bool isMinus;//是否为负数

int size;//链表长度

node \*first;//头指针

node \*last;//尾指针

};

#endif

## interface.h

//interface.h

void mainface();

void addface();

void subtractface();

void multiplyface();

void divideface();

void powerface();

void file\_add();

void file\_subtract();

void file\_mutiply();

void file\_divideface();

void file\_power();

void keyboard\_add();

void keyboard\_subtract();

void keyboard\_mutiply();

void keyboard\_divide();

void keyboard\_power();

## bigInteger.cpp

//bigInteger.cpp

#include<iostream>

#include<string>

#include"bigInteger.h"

using namespace std;

//add

linkedList bigInteger::add(const linkedList &left,const linkedList &right)

{

linkedList tmp;//新建一个空链表

int tempNum,rest=0;//tempNum是每一位的结果，rest是进位的数值

node \*n1=left.last;

node \*n2=right.last;

//两数一样长

while(n1&&n2)

{

tempNum=n1->data+n2->data+rest;

if(tempNum>9)

{

tempNum=tempNum-10;

rest=1;

}

else

rest=0;

tmp.push\_front(tempNum);

n1=n1->pre;

n2=n2->pre;

}

//上面的数比下面短

while(n1)

{

tempNum=n1->data+rest;

if(tempNum>9)

{

tempNum=tempNum-10;

rest=1;

}

else

rest=0;

tmp.push\_front(tempNum);

n1=n1->pre;

}

//下面的数比上面短

while(n2)

{

tempNum=n2->data+rest;

if(tempNum>9)

{

tempNum=tempNum-10;

rest=1;

}

else

rest=0;

tmp.push\_front(tempNum);

n2=n2->pre;

}

//如果最后一次进位了，前面加1

if(rest)

tmp.push\_front(1);

return tmp;

}

//比大小：1是前面大，2是后面大，0是一样大

int bigInteger::isLarge(const linkedList &left,const linkedList &right)

{

if(left.size>right.size)

return 1;

else if(left.size<right.size)

return 2;

node \*p1=left.first;

node \*p2=right.first;

for(int i=0;i<left.size;i++)

{

if(p1->data>p2->data)

{

return 1;

}

else if(p1->data<p2->data)

{

return 2;

}

p1=p1->next;

p2=p2->next;

}

return 0;

}

//subtract

linkedList bigInteger::subtract(const linkedList &left,const linkedList &right)

{

linkedList tmp;//新建一个空链表

int tempNum,rest=0;//tempNum是每一位的结果，rest是进位的数值

node \*n1=left.last;

node \*n2=right.last;

int whoLarge=isLarge(left,right);

//一样大，结果为0

if(whoLarge==0)

tmp.push\_front(0);

//上面的数大

if(whoLarge==1)

{

//上下的数都有

while(n1&&n2)

{

tempNum=n1->data-n2->data+rest;

if(tempNum<0)

{

tempNum=tempNum+10;

rest=-1;

}

else

rest=0;

tmp.push\_front(tempNum);

n1=n1->pre;

n2=n2->pre;

}

//下面的数没了

while(n1)

{

tempNum=n1->data+rest;

if(tempNum<0)

{

tempNum=tempNum+10;

rest=-1;

}

else

rest=0;

tmp.push\_front(tempNum);

n1=n1->pre;

}

}

//后者大

else if(whoLarge==2)

{

tmp.isMinus=true;

//上下的数都有

while(n2&&n1)

{

tempNum=n2->data-n1->data+rest;

if(tempNum<0)

{

tempNum=tempNum+10;

rest=-1;

}

else

rest=0;

tmp.push\_front(tempNum);

n1=n1->pre;

n2=n2->pre;

}

//下面的数没了

while(n2)

{

tempNum=n2->data+rest;

if(tempNum<0)

{

tempNum=tempNum+10;

rest=-1;

}

else

rest=0;

tmp.push\_front(tempNum);

n2=n2->pre;

}

}

//去掉前面的0

while(tmp.first->data==0)

{

if(tmp.size==1)

break;

node \*temp=tmp.first;

tmp.first=tmp.first->next;

delete temp;

tmp.size--;

}

return tmp;

}

//multiply

linkedList bigInteger::multiply(const linkedList &left,const linkedList &right)

{

linkedList tmp;//新建一个空链表

int tempNum=0,rest=0;//tempNum是每一位的结果，rest是进位的数值

node \*n1=left.last;

node \*n2=right.last;

node \*result=0;

node \*tempResult=0;

bool isTmpEmpty=true;//哨兵

//两个数中有一个是0的情况

if(left.size==1&&n1->data==0||right.size==1&&n2->data==0)

{

tmp.push\_front(0);

return tmp;

}

//两个数中有一个是1的情况

else if(left.size==1&&n1->data==1||right.size==1&&n2->data==1)

{

if(n1->data==1)

tmp=right;

else

tmp=left;

return tmp;

}

else if(left.size>=right.size)

{

//先求卷积的结果

while(n2)

{

tempResult=result;

int flag=0;

while(n1)

{

flag++;

tempNum=n2->data\*n1->data;

if(tempResult!=0)

{

tempNum=tempResult->data+tempNum;

tempResult->data=tempNum;

tempResult=tempResult->pre;

n1=n1->pre;

}

else

{

tmp.push\_front(tempNum);

if(isTmpEmpty)

{

result=tmp.last;

tempResult=result;

isTmpEmpty=false;

}

n1=n1->pre;

for(int i=0;i<flag;i++)

{

if(i==0)

tempResult=result->pre;

else

tempResult=tempResult->pre;

}

}

}

n1=left.last;

result=result->pre;

n2=n2->pre;

}

}

else

{

//先求卷积的结果

while(n1)

{

tempResult=result;

int flag=0;

while(n2)

{

flag++;

tempNum=n1->data\*n2->data;

if(tempResult!=0)

{

tempNum=tempResult->data+tempNum;

tempResult->data=tempNum;

tempResult=tempResult->pre;

n2=n2->pre;

}

else

{

tmp.push\_front(tempNum);

if(isTmpEmpty)

{

result=tmp.last;

tempResult=result;

isTmpEmpty=false;

}

n2=n2->pre;

for(int i=0;i<flag;i++)

{

if(i==0)

tempResult=result->pre;

else

tempResult=tempResult->pre;

}

}

}

n2=left.last;

result=result->pre;

n1=n1->pre;

}

}

node \*p=tmp.last;

while(p)

{

tempNum=p->data+rest;

if(tempNum>9)

{

rest=tempNum/10;

tempNum=tempNum%10;

}

else

rest=0;

p->data=tempNum;

p=p->pre;

}

if(rest!=0)

tmp.push\_front(rest);

return tmp;

}

//divide

linkedList bigInteger::divide(const linkedList &left,const linkedList &right)

{

linkedList quotient;//商

linkedList remainder;//余数

int whoLarge=isLarge(left,right);//比大小

//一样大，结果为1

if(whoLarge==0)

{

quotient.push\_front(1);

remainder.push\_front(0);

return quotient;//mod则return remainder;

}

//后者大，结果为0

else if(whoLarge==2)

{

quotient.push\_front(0);

remainder=left;

return quotient;//mod则return remainder;

}

//除数是1

else if(right.size==1&&right.first->data==1)

{

quotient=left;

remainder.push\_front(0);

return quotient;//mod则return remainder;

}

//普通情况

else

{

node \*p=0;

//截取被除数中少于除数1位的链表作为起始的余数

if(right.size-1!=0)

{

remainder.create(right.size-1);

node \*p1=remainder.first;

node \*p2=left.first;

//循环赋值

for(int i=0;i<right.size-1;i++)

{

p1->data=p2->data;

p1=p1->next;

p2=p2->next;

}

}

node \*tmp=left.first;

//tmp跳到下一位准备运算

for(int j=0;j<right.size-1;j++)

tmp=tmp->next;

for(int i=right.size-1;i<left.size;i++)

{

p=remainder.first;

//相减之后差为0,将0后的数代替0

if(p!=0&&p->data==0)

{

node \*change=remainder.first;

change->data=tmp->data;

}

//继续插入下一位数

else

remainder.push\_back(tmp->data);

//一位一位的计算商

for(int tmpQuotient=0;whoLarge=isLarge(remainder,right)!=2;tmpQuotient++)

remainder=subtract(remainder,right);

tmp=tmp->next;

//将循环次数作为商放进结果

quotient.push\_back(tmpQuotient);

}

//去掉每次的余数前面的0（针对上次计算后余数为0的情况）

if(quotient.first->data==0)

{

node \*temp=quotient.first;

quotient.first=quotient.first->next;

delete temp;

quotient.size--;

}

return quotient;//mod则return remainder;

}

}

//mod

linkedList bigInteger::mod(const linkedList &left,const linkedList &right)

{

linkedList quotient;//商

linkedList remainder;//余数

int whoLarge=isLarge(left,right);//比大小

//一样大，结果为1

if(whoLarge==0)

{

quotient.push\_front(1);

remainder.push\_front(0);

return remainder;//mod则return remainder;

}

//后者大，结果为0

else if(whoLarge==2)

{

quotient.push\_front(0);

remainder=left;

return remainder;//mod则return remainder;

}

//除数是1

else if(right.size==1&&right.first->data==1)

{

quotient=left;

remainder.push\_back(0);

return remainder;//mod则return remainder;

}

//普通情况

else

{

node \*p=0;

//截取被除数中少于除数1位的链表作为起始的余数

if(right.size-1!=0)

{

remainder.create(right.size-1);

node \*p1=remainder.first;

node \*p2=left.first;

//循环赋值

for(int i=0;i<right.size-1;i++)

{

p1->data=p2->data;

p1=p1->next;

p2=p2->next;

}

}

node \*tmp=left.first;

//tmp跳到下一位准备运算

for(int j=0;j<right.size-1;j++)

tmp=tmp->next;

for(int i=right.size-1;i<left.size;i++)

{

p=remainder.first;

//相减之后差为0,将0后的数代替0

if(p!=0&&p->data==0)

{

node \*change=remainder.first;

change->data=tmp->data;

}

//继续插入下一位数

else

remainder.push\_back(tmp->data);

//一位一位的计算商

for(int tmpQuotient=0;whoLarge=isLarge(remainder,right)!=2;tmpQuotient++)

remainder=subtract(remainder,right);

tmp=tmp->next;

//将循环次数作为商放进结果

quotient.push\_back(tmpQuotient);

}

//去掉每次的余数前面的0（针对上次计算后余数为0的情况）

if(quotient.first->data==0)

{

node \*temp=quotient.first;

quotient.first=quotient.first->next;

delete temp;

quotient.size--;

}

return remainder;//mod则return remainder;

}

}

//power

linkedList bigInteger::power(const linkedList &left,const linkedList &right)

{

linkedList tmp;

//将做指数运算的数是1或者0

if(left.size==1&&(left.last->data==1||left.last->data==0))

{

//1

if(left.last->data==1)

tmp.push\_front(1);

//0

else

tmp.push\_front(0);

return tmp;

}

//指数为1或者0

else if(right.size==1&&(right.last->data==1||right.last->data==0))

{

//指数为1

if(right.last->data==1)

{

linkedList temp(left);

return temp;

}

//指数为0

else

{

tmp.push\_front(1);

return tmp;

}

}

else

{

linkedList zero,one;

zero.push\_front(0);

one.push\_front(1);

linkedList tmpResult;

tmpResult.push\_front(1);

/\*

\*下面的循环是根据普通的指数运算改成的

\*假设计算a^b;

\*int tmpResult=1;

\*for(int i=次数;i>0;i--)

\* tmpResult=tmpResult\*a;

\*/

for(linkedList exponent(right);isLarge(exponent,zero)==1;exponent=subtract(exponent,one))

tmpResult=multiply(tmpResult,left);

return tmpResult;

}

}

## linkedList.cpp

//linkedList.cpp

#include<iostream>

#include<string>

#include<fstream>

#include"linkedList.h"

using namespace std;

//重载默认构造函数

linkedList::linkedList()

{

size=0;

isMinus=false;

first=last=0;

}

//创建指定长度的链表，并初始化为0

void linkedList::create(int lsize)

{

size=lsize;

isMinus=false;

for(int i=0;i<size;i++)

{

if(i==0)

{

first=new node(0,0,0);

last=first;

}

else

{

node \*temp=new node(0,0,0);

temp->pre=last;

last->next=temp;

last=last->next;

}

}

}

//构造函数

linkedList::linkedList(string &s)

{

isMinus=false;

//去掉前面的0

if(s[0]=='0')

{

//开头最后一个0的下标

int num=s.length()-1;

for(int i=1;i<s.length()-1;i++)

{

if(s[i]!='0')

{

num=i;

break;

}

}

s=s.substr(num,s.length()-num);

}

//链表长度与字符串长度相同

size=s.length();

for(int i=0;i<size;i++)

{

if(i==0)

{

first=new node(s[i]-'0',0,0);

last=first;

}

else

{

node \*temp=new node(s[i]-'0',0,0);

temp->pre=last;

last->next=temp;

last=last->next;

}

}

}

//析构函数：释放动态申请的空间

linkedList::~linkedList()

{

while(first!=0)

{

node \*temp=first;

first=first->next;

delete temp;

}

}

//清空

void linkedList::clear()

{

while(first!=0)

{

node \*temp=first;

first=first->next;

delete temp;

}

}

//拷贝构造函数

linkedList::linkedList(const linkedList &l)

{

isMinus=l.isMinus;

//给空链表拷贝一个空链表

if(l.first==0)

{

first=0;

size=0;

}

else

{

last=0;

node \*temp=l.first;

for(int i=0;temp!=0;i++)

{

if(i==0)

{

first=new node(temp->data,0);

last=first;

temp=temp->next;

}

else

{

node \*tmp=new node(temp->data,0);

tmp->pre=last;

last->next=tmp;

last=last->next;

temp=temp->next;

}

}

size=i;

}

}

//重载=

linkedList& linkedList::operator=(const linkedList& rhs)

{

if(this==&rhs)

return \*this;

clear();

isMinus=rhs.isMinus;

//给空链表拷贝一个空链表

if(rhs.first==0)

{

first=0;

size=0;

}

else

{

last=0;

node \*temp=rhs.first;

for(int i=0;temp!=0;i++)

{

if(i==0)

{

first=new node(temp->data,0);

last=first;

temp=temp->next;

}

else

{

node \*tmp=new node(temp->data,0);

tmp->pre=last;

last->next=tmp;

last=last->next;

temp=temp->next;

}

}

size=i;

}

return \*this;

}

//输出链表函数

void linkedList::display()

{

if(isMinus)

cout<<'-';

int reminder=size%3;

node \*temp=first;

if(reminder==0)

{

for(int j=1;j<=size-1;j++)

{

cout<<temp->data;

temp=temp->next;

if(j%3==0)

cout<<',';

}

cout<<temp->data;

cout<<endl;

cout<<"共计 "<<size<<" 位。"<<endl;

}

else

{

for(int i=0;i<size-1;i++)

{

cout<<temp->data;

temp=temp->next;

if(i==reminder-1)

{

cout<<',';

reminder+=3;

}

}

cout<<temp->data;

cout<<endl;

cout<<"共计 "<<size<<" 位。"<<endl;

}

}

//导出链表

void linkedList::savetxt(const string &path)

{

ofstream fout;

fout.open(path.c\_str());

if(isMinus)

fout<<'-';

int reminder=size%3;

node \*temp=first;

if(reminder==0)

{

for(int j=1;j<=size-1;j++)

{

fout<<temp->data;

temp=temp->next;

if(j%3==0)

fout<<',';

}

fout<<temp->data;

fout<<endl;

fout<<"共计 "<<size<<" 位。"<<endl;

}

else

{

for(int i=0;i<size-1;i++)

{

fout<<temp->data;

temp=temp->next;

if(i==reminder-1)

{

fout<<',';

reminder+=3;

}

}

fout<<temp->data;

fout<<endl;

fout<<"共计 "<<size<<" 位。"<<endl;

}

fout.close();

}

//在头部添加一个元素

void linkedList::push\_front(const int &item)

{

node \*p=0;

node \*tmp=0;

p=new node(item,0,0);

//如果是空链表

if(first==0)

{

p->next=0;

p->pre=0;

last=p;

first=p;

}

//非空链表，则把旧元素的pre指向p,并将p的next指向原来的头元素

else

{

tmp=first;

first=p;

p->next=tmp;

tmp->pre=p;

p->pre=0;

}

size++;

}

//在尾部添加一个元素

void linkedList::push\_back(const int &item)

{

node \*p=0;

node \*tmp=0;

p=new node(item,0,0);

//如果是空链表

if(first==0)

{

p->next=0;

p->pre=0;

last=p;

first=p;

}

else

{

tmp=last;

last=p;

p->next=0;

tmp->next=p;

p->pre=tmp;

}

size++;

}

## interface.cpp

//interface.cpp

#include<iostream>

#include<string>

#include<ctime>

#include<fstream>

#include<windows.h>

#include"bigInteger.h"

#include"linkedList.h"

#include"interface.h"

using namespace std;

void file\_add()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 加 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请输入文本路径："<<endl;

cout<<"例如：D:/xxx/xxx.txt"<<endl<<endl;

bigInteger para;

string path,n1,n2;

while(cin>>path)

{

if(path=="R")

addface();

//导入文本

ifstream fin;

fin.open(path.c\_str());

if(!fin)

{

cout<<"路径有误，请重新输入！"<<endl<<endl;

continue;

}

else

fin>>n1>>n2;

fin.close();

break;

}

cout<<" 正 在 读 入 文 本"<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(10);

}

cout<<endl;

cout<<" 完 毕"<<endl<<endl;

cout<<"第一个数："<<n1<<endl<<endl;

cout<<"第二个数："<<n2<<endl<<endl;

linkedList l1(n1);

linkedList l2(n2);

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

addface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//加的结果

result=para.add(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//加的结果

linkedList l5=para.add(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

addface();

else if(choice=='Y')

continue;

}

}

void keyboard\_add()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 加 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

bigInteger para;

string n1,n2;

cout<<"请输入第一个数：";

cin>>n1;

if(n1=="R")

addface();

linkedList l1(n1);

cout<<endl;

cout<<"请输入第二个数：";

cin>>n2;

if(n2=="R")

addface();

linkedList l2(n2);

cout<<endl;

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

linkedList result;

double start,end;

cin>>num>>expo;

if(num=="R"||expo=="R")

addface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

result=para.add(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//加的结果

linkedList l5=para.add(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

addface();

else if(choice=='Y')

continue;

}

}

void addface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 加 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择输入方式：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':keyboard\_add();break;

case '2':file\_add();break;

case 'R':mainface();break;

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

void file\_subtract()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 减 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请输入文本路径："<<endl;

cout<<"例如：D:/xxx/xxx.txt"<<endl<<endl;

bigInteger para;

string path,n1,n2;

while(cin>>path)

{

if(path=="R")

subtractface();

//导入文本

ifstream fin;

fin.open(path.c\_str());

if(!fin)

{

cout<<"路径有误，请重新输入！"<<endl<<endl;

continue;

}

else

fin>>n1>>n2;

fin.close();

break;

}

cout<<" 正 在 读 入 文 本"<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(10);

}

cout<<endl;

cout<<" 完 毕"<<endl<<endl;

cout<<"第一个数："<<n1<<endl<<endl;

cout<<"第二个数："<<n2<<endl<<endl;

linkedList l1(n1);

linkedList l2(n2);

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

subtractface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//减的结果

result=para.subtract(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//减的结果

linkedList l5=para.subtract(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

subtractface();

else if(choice=='Y')

continue;

}

}

void keyboard\_subtract()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 减 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

bigInteger para;

string n1,n2;

cout<<"请输入第一个数：";

cin>>n1;

if(n1=="R")

subtractface();

linkedList l1(n1);

cout<<endl;

cout<<"请输入第二个数：";

cin>>n2;

if(n2=="R")

subtractface();

linkedList l2(n2);

cout<<endl;

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

subtractface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//减的结果

result=para.subtract(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//减的结果

linkedList l5=para.subtract(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

subtractface();

else if(choice=='Y')

continue;

}

}

void subtractface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 减 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择输入方式：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':keyboard\_subtract();break;

case '2':file\_subtract();break;

case 'R':mainface();break;

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

void file\_multiply()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 乘 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请输入文本路径："<<endl;

cout<<"例如：D:/xxx/xxx.txt"<<endl<<endl;

bigInteger para;

string path,n1,n2;

while(cin>>path)

{

if(path=="R")

multiplyface();

//导入文本

ifstream fin;

fin.open(path.c\_str());

if(!fin)

{

cout<<"路径有误，请重新输入！"<<endl<<endl;

continue;

}

else

fin>>n1>>n2;

fin.close();

break;

}

cout<<" 正 在 读 入 文 本"<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(10);

}

cout<<endl;

cout<<" 完 毕"<<endl<<endl;

cout<<"第一个数："<<n1<<endl<<endl;

cout<<"第二个数："<<n2<<endl<<endl;

linkedList l1(n1);

linkedList l2(n2);

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

multiplyface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//乘的结果

result=para.multiply(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//乘的结果

linkedList l5=para.multiply(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

multiplyface();

else if(choice=='Y')

continue;

}

}

void keyboard\_multiply()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 乘 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

bigInteger para;

string n1,n2;

cout<<"请输入第一个数：";

cin>>n1;

if(n1=="R")

multiplyface();

linkedList l1(n1);

cout<<endl;

cout<<"请输入第二个数：";

cin>>n2;

if(n2=="R")

multiplyface();

linkedList l2(n2);

cout<<endl;

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

multiplyface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//乘的结果

result=para.multiply(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//乘的结果

linkedList l5=para.multiply(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时："<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

multiplyface();

else if(choice=='Y')

continue;

}

}

void multiplyface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 乘 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择输入方式：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':keyboard\_multiply();break;

case '2':file\_multiply();break;

case 'R':mainface();break;

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

void file\_divide()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 除 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请输入文本路径："<<endl;

cout<<"例如：D:/xxx/xxx.txt"<<endl<<endl;

bigInteger para;

string path,n1,n2;

while(cin>>path)

{

if(path=="R")

divideface();

//导入文本

ifstream fin;

fin.open(path.c\_str());

if(!fin)

{

cout<<"路径有误，请重新输入！"<<endl<<endl;

continue;

}

else

fin>>n1>>n2;

fin.close();

break;

}

cout<<" 正 在 读 入 文 本"<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(10);

}

cout<<endl;

cout<<" 完 毕"<<endl<<endl;

cout<<"第一个数："<<n1<<endl<<endl;

cout<<"第二个数："<<n2<<endl<<endl;

linkedList l1(n1);

linkedList l2(n2);

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

divideface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//除的结果

result=para.divide(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//除的结果

linkedList l5=para.divide(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

divideface();

else if(choice=='Y')

continue;

}

}

void keyboard\_divide()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 除 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

bigInteger para;

string n1,n2;

cout<<"请输入第一个数：";

cin>>n1;

if(n1=="R")

divideface();

linkedList l1(n1);

cout<<endl;

cout<<"请输入第二个数：";

cin>>n2;

if(n2=="R")

divideface();

while(n2=="0")

{

cout<<"除数不能为0！"<<endl<<endl;

cout<<"请输入第二个数：";

cin>>n2;

}

linkedList l2(n2);

cout<<endl;

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

divideface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//除的结果

result=para.divide(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//除的结果

linkedList l5=para.divide(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时："<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

divideface();

else if(choice=='Y')

continue;

}

}

void divideface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 除 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择输入方式：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':keyboard\_divide();break;

case '2':file\_divide();break;

case 'R':mainface();break;

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

void file\_power()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 指 数 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请输入文本路径："<<endl;

cout<<"例如：D:/xxx/xxx.txt"<<endl<<endl;

bigInteger para;

string path,n1,n2;

while(cin>>path)

{

if(path=="R")

powerface();

//导入文本

ifstream fin;

fin.open(path.c\_str());

if(!fin)

{

cout<<"路径有误，请重新输入！"<<endl<<endl;

continue;

}

else

fin>>n1>>n2;

fin.close();

break;

}

cout<<" 正 在 读 入 文 本"<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(10);

}

cout<<endl;

cout<<" 完 毕"<<endl<<endl;

cout<<"第一个数："<<n1<<endl<<endl;

cout<<"第二个数："<<n2<<endl<<endl;

linkedList l1(n1);

linkedList l2(n2);

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

powerface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//幂的结果

result=para.power(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//幂的结果

linkedList l5=para.power(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时： "<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

powerface();

else if(choice=='Y')

continue;

}

}

void keyboard\_power()

{

char choice;

while(true)

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 指 数 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

bigInteger para;

string n1,n2;

cout<<"请输入第一个数：";

cin>>n1;

if(n1=="R")

powerface();

while(n1=="0")

{

cout<<"底数不能为0！"<<endl<<endl;

cout<<"请输入第一个数：";

cin>>n1;

}

linkedList l1(n1);

cout<<endl;

cout<<"请输入第二个数：";

cin>>n2;

if(n2=="R")

powerface();

linkedList l2(n2);

cout<<endl;

cout<<"提示：1.底数和指数以空格隔开。"<<endl;

cout<<" 2.输入0 0即可跳过此步。"<<endl;

cout<<"请输入指数形式的模数：";

string num,expo;

double start,end;

linkedList result;

cin>>num>>expo;

if(num=="R"||expo=="R")

powerface();

if(num=="0"&&expo=="0")

{

cout<<endl;

start=clock();

//幂的结果

result=para.power(l1,l2);

end=clock();

}

else

{

linkedList l3(num);

linkedList l4(expo);

cout<<endl;

start=clock();

//幂的结果

linkedList l5=para.power(l1,l2);

//幂的结果

linkedList l6=para.power(l3,l4);

result=para.mod(l5,l6);

end=clock();

}

cout<<"结果为：";

result.display();

cout<<endl;

cout<<"用时："<<(end-start)/1000<<" 秒"<<endl<<endl;

while(true)

{

cout<<"是否导出至文件？(Y/N)：";

char choice2;

while(cin>>choice2)

{

if(choice2=='Y')

{

cout<<"例如：D:/xxx/xxx.txt"<<endl;

cout<<"注：1.若文本不存在则创建该文本。"<<endl;

cout<<" 2.若文本存在则会替换文本的内容。"<<endl<<endl;

cout<<"请输入目的路径：";

string path;

cin>>path;

result.savetxt(path);

cout<<"导出成功。"<<endl<<endl;

break;

}

else if(choice2=='N')

{

cout<<endl;

break;

}

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

cout<<"是否继续输入？(Y/N)：";

cin>>choice;

cout<<endl;

if(choice=='N'||choice=='Y')

break;

else

cout<<"指令有误，请重新输入！"<<endl<<endl;

}

if(choice=='N')

powerface();

else if(choice=='Y')

continue;

}

}

void powerface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 指 数 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.键 盘 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.文 件 输 入 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入R以返回 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择输入方式：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':keyboard\_power();break;

case '2':file\_power();break;

case 'R':mainface();break;

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

void mainface()

{

system("cls");

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 1.加 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 2.减 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 3.乘 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 4.除 法 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 5.指 数 运 算 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 输入Q以退出 \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<"请选择：";

char choice;

while(cin>>choice)

{

switch(choice)

{

case '1':addface();break;

case '2':subtractface();break;

case '3':multiplyface();break;

case '4':divideface();break;

case '5':powerface();break;

case 'Q':exit(0);

default :

cout<<"指令有误，请重新输入！"<<endl;

}

}

}

## main.cpp

//main.cpp

#include<iostream>

#include<windows.h>

#include"bigInteger.h"

#include"linkedList.h"

#include"interface.h"

using namespace std;

void main()

{

cout<<endl;

cout<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \*大 整 数 运 算(Alpha版)\* \* \* \* \* \* \* \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 作 者： \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 朱润石 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* 胡莎莎 \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \*"<<endl;

cout<<" \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*"<<endl;

cout<<endl;

cout<<" STARTING "<<endl;

cout<<endl;

cout<<" ";

for(int i=0;i<61;i++)

{

cout<<">";

Sleep(50);

}

cout<<endl;

mainface();

}

# 开发人员

**Rush Chuh (Zhejiang University of Technology)**