ASE 实验报告

本次实验面对最大的问题就是存储限制。

Project 1：万进制基础上的大数运算

警告：

C完成，如果不能编译尝试使用 /D \_CRT\_SECURE\_NO\_WARNINGS

自己编写的时候输入是从txt读入的，限于数据结构，不一定能成功接收输入，待完善~

除法非常吃内存，建议开64位并关注内存使用情况，随时关闭~

数据结构：

选择单链表作为数据结构，每一个节点上存储一个4位的int型，即万进制数；

位数是右对齐的，在读入时调整，最终得到的是一个右对齐，倒序的万进制链表；

采用4位万进制，21组卷积均是基于编译器规定int型为4个字节，如果采用字长更长的变量进行操作，计算效率将大大提升。

算法：

+，-操作与手动算法无异，加法逢万进位，减法不足借万；

减法返回的是绝对差值，如果需要返回负差值，则不做最后的调整即可；

\*操作采用21项卷积21项的类多项式乘法完成，即固定一条链上的21组连续万进制数，以21组为长度遍历另一条链，结果相加；

/操作模拟辗转相除法，预估商的位数和满足右对齐要求的位数的商的前K位的值，取不超过她的K位数，构造商；

性能：

加法和减法都是毫秒级；

乘法65万位乘以65万位是67秒，60万位乘30万位是25秒，这是改变卷积组数（从24分钟到67秒）优化得出的结构，如果字长允许，算法的效率也应该是不错的；

除法对内存有一定吞噬，问题在查找中~如果能算，大概几十秒的速度；如果内存消耗，也会很快出错；

Project 2：筛法基础上的素数查找

数据结构：

这个程序是一个简单的筛法，主要问题在内存不足，C无法申请到我需要的空间大小，最终选择C++的布尔型变量来存储；

算法：

本身在C里面写的是欧拉筛法，速度较快，但限于内存放弃；

实际采用普通筛法，做优化：如越过所有的偶数；如只需测试到i<sqrt(m),但是因为精度的问题，其实不如用i\*i<m作为判断条件；如越过已经筛过一次的数；

其实欧拉的优化在于假设某一个数可以被第j个素数整除，而素数是递增的，应该跳出继续筛选的过程避免重复筛，但是缺点在于需要保存下各个素数的值，可是无法申请到如此多连续的int空间，只得作罢；

性能：

测试正常，十几秒左右；

IM 组

冯晓云