实验 5 线性表应用——大整数运算

一、实验目的

巩固和加深对线性表的理解，希望同学们能够熟练掌握顺序表、链表的基本操作、能够应用线性表解决实际问题。

二、预备知识

1、运算顺序：两个大整数靠右对齐；从低位向高位运算；先计算低位再计算高位。

2、运算规则：如加法运算，同一位的两个数相加再加上从低位来的进位，成为该位的和；该和去掉向高位的进位就成为该位的值；如：3+8+1=12，向前一位进 1，本位值是 2，可借助‘%’、‘/’运算完成。

3、最后一位的进位：若完成两数相加后，进位位值不为 0，则应添加一位。

4、带符号的大整数运算可以借助无符号的大整数运算。如：负整数+正整数，可看成两个无符号整数的减法运算。

5、大整数的范围大大超出了标准数据类型（整型、实型）能表示的范围，所以需要借助相应的数据结构来表示，如：线性表，可以采用链式或顺序存储方式。通过分析这两种结构各有的优缺点，从而选择合适的存储结构。

(1) 链式存储适应不定长度的大整数，存储空间包括大整数的表示部分和指针部分，其空间利用率不高，随机访问效率低。

(2) 顺序存储，其元素个数不能自由扩充，在运算时易造成溢出，但可随机访问，且空间利用相对较高。

三、实验题目——实现大整数的运算

1、基本要求

(1) 大整数的长度在 30 位以上；

(2) 考虑正负数。

2、主要功能

(1) 实现大整数的输入；

(2) 实现大整数的输出；

(3) 实现比较两个大整数的大小关系；

(4) 实现大整数的加法运算；

(5) 实现大整数的减法运算；

(6) 实现大整数的乘法运算；

(7) 实现大整数的除法运算（**选做**）；

(8) 通过分析算法的时间复杂度和空间复杂度，对算法效率进行优化，并实现（**选做**）。

四、实验要求

(1) 大整数存储结构的定义和基本操作函数声明放在 BigNum.h 文件；

(2) 基本功能的实现放在单独的 BigNum.c 文件；

(3) 测试程序放在 BigNumTestApp.c 中。