程序设计基础课程设计报告

——高精度计算

高宇轩 23009200132

2024 年 4 月 18 日

1 原始题目及要求

用整型数组表示 10 进制大整数（超过 2^{32} 的整数），数组的每个元素存储大整数的一位数字，实现大整数的加减法。

2 题目分析

2.1 题目功能

对于超过 2^{32} 的整数，超出了 int 能够存储的最大值，此时需要用整型数组来表示十进制的整数，我们可以用模拟竖式加减法的方式对数组进行运算。

2.2 题目知识点

数组、流程控制、函数等

3 题目总体方案设计

3.1 程序包含的模块结构图

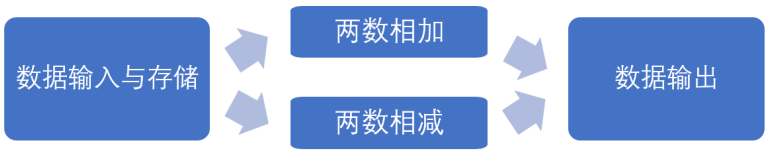

图 1: 程序的模块结构图

3.2 输入输出数据说明

输入数据：两个任意大小，可正可负可为零的整数
输出数据：分别输出这两个整数的和与差

3.3 数据结构说明

将大整数存入 bigInt 类中，用一个 bool 变量维护整数的正负，用一个 vector<int> 存储整数的每一个数位

4 各功能模块的设计说明

4.1 整数输入

先将输入的大整数存入字符串中，根据字符串首位是否为'-' 判断整数的正负，然后将整数按位依次存入 vector<int> 中

4.2 整数相加

先判断两个数的符号，如果都为正或都为负，那么直接将两个数的绝对值相加，并且答案的符号与两个数相同；如果两个数为一正一负，那么用第一个数减去负的第二个数，此时问题转化为两个正数或者两个负数相减。

在将两个数的绝对值相加时，从低位至高位模拟两个数的竖式加法，如果同位的两个数相加大于十，那么答案的这一位为两个数相加减十，同时向下一位的加法进位，如果出现两个数长度不同，则用零补足，直到全部加完。

4.3 整数相减

同样先判断两个数的符号，如果都为正或都为负，那么先判断结果的正负，然后用大的绝对值减去小的绝对值；如果两个数为一正一负，那么用第一个数加上负的第二个数，此时问题转化为两个正数或者两个负数相加。

在用大的绝对值减去小的绝对值时，从低位至高位模拟两个数的竖式减法，如果相减小于零，从前一位借位，如果出现两个数长度不同，则用零补足，直到全部减完。

4.4 整数输出

重载 « 运算符，先将 bigInt 转化为字符串，然后输出字符串。

将 bigInt 转化为字符串时，先将数组中的数从高位至低位放入字符串中，同时用一个 bool 值判断是否为前导零。将数写入字符串后，如果字符串为空，把字符串赋值为零字符，如果不为空且有负号，在字符串前加上负号，最后输出这个字符串。

5 程序的集成测试

```
Ubuntu: '/mnt/e/202402-202406/C++Programs/23009200132/1BigInt/cmake-build-debug/BigInt'
3 + 2 = 5
3 - 2 = 1

2 + 3 = 5
2 - 3 = -1

1234567890 + 987654321 = 2222222211
1234567890 - 987654321 = 246913569

-9999999999999999 + 1 = -9999999999999998
-9999999999999999 - 1 = -10000000000000000

-246810 + -13579 = -260389
-246810 - -13579 = -233231

114514 + -1919810 = -1805296
114514 - -1919810 = 2034324

0 + 0 = 0
0 - 0 = 0
```

图 2: 程序的集成测试

6 总结

在集成测试中，我们分别测试了两个正数，一个正数与一个负数，一个负数与一个正数，两个负数，以及两个都为零的特殊情况，可以看出程序的运行结果都是符合我们预期的，实现了题目的要求。

在今后的改进中，我们还可以用模拟整数乘法的方式，尝试编写两个大整数的乘法，进一步扩展我们的 bigInt 类。