面向对象程序设计基础作业十 设计文档

1. 模型部分
   1. 功能简述

本程序实现了大整数的加减法，且同时支持正、负整数，理论上无上限（实际受限于栈空间大小）。用户需要输入两个整数和一个+或-号，并用空格分割，程序会输出对应的结果。

* 1. 算法

采用整数数组模拟竖式过程，逐位存储大整数每一位的数值，并额外增加一布尔型变量用以存储大整数的符号。其中，0的符号规定为正负均可，在输出时会自动过滤（不输出负号）。

对于加法，我们先判断每个加数是否为负数，若是则暂时将每一位置为原来的相反数。之后，从最低位起，将两个加数的每一位相加，并检查是否小于0或大于9。如果小于0，则将下一位减一、本位加十；如果大于9，则将下一位加一、本位减十。如果最高位小于0，则将整个数置为负数，然后再执行一遍退位操作（相当于用10n减去此数）。

对于减法，我们直接调用加法，加上被减数的相反数即可。

1. 验证部分
   1. 测试数据说明

测试所用数据为：

* + 1. 34 1267: 代表正整数，含进位
    2. -34 1267: 代表第一个数是负数，第二个数是正数
    3. 34 -1267: 代表第一个数是正数，第二个数是负数
    4. -34 -1267: 代表两个负数
    5. 0 0: 代表全为零
    6. 42 -42: 代表两个互为相反数的数，测试0的输出情况
    7. 165276347677824857 2136756878984346782：测试对大整数的支持（网页计算器显示其和为2302033226662171639）
  1. 测试结果

|  |  |  |
| --- | --- | --- |
| 测试数据 | 相加 | 相减 |
| 34 1267 | 1301 | -1233 |
| -34 1267 | 1233 | -1301 |
| 34 -1267 | -1233 | 1301 |
| -34 -1267 | -1301 | 1233 |
| 0 0 | 0 | 0 |
| 42 -42 | 0 | 84 |
| 最后一组大数 | 2302033226662171639 | -1971480531306521925 |

受空间所限，**部分**具有代表性的测试截图如下：

