C++基础与深度解析

各位同学们，大家好，相信你们早已经摩拳擦掌等待第一个项目的挑战了。接下来，是王者，还是青铜，该现原形了。

项目 1：长整数加法

一个文件中包含了两行数字序列，每行数字序列表示一个十进制长整数（只包含 0~9 与符号位，如果存在符号位，那么只能处于开头位置）。你的程序应当接收一个输入参数，表示该文件，程序读取这两个序列并求二者之和，将结果输出到屏幕上。

需要注意的是，由于序列可能比较长，因此其对应的数值可能无法使用一个内建数据类型来保存。需要使用一个数组（或 vector）来保存每一位的数值。

注意事项：

1. 整数前面可能包含表示正数的'+'符号。你的程序应当允许这种输入，但输出结果中不应该包含该符号。
2. 注意如果两个数的符号位不同，那么实际上实现的应当是减法。
3. 从数学上来说，一个十进制数前面如果加 0 不会影响该数的值，比如 123 与 0123 表示的是相同的值。你的程序在读取数据时应当能够处理这种情况，但输出结果时不应包含前面位置的 0。

扩展 1：

程序可以再接收一个表示进制的参数，以实现不同进制下的长整数加法，比如：

MyProm InputFile 8

实现的就是两个 8 进制长整数相加。第二个如果大于 10，那么就可以使用字母来表示相应的值，比如使用 a 或 A 来表示十进制的 10，使用 b 或 B 来表示十进制的 11，依此类推（参考 16 进制的表示方式）。

举例来说，对于一个 20 进制的整数：1a3 所对应的十进制为：1*400 + 10*20 +3 = 603。

注意：进制最多支持到 36，使用 0~9,a~z（或 A~Z）共 36 个字符表示。如果用户输入的第二个参数不是一个合法的值，那么程序应当返回相应的错误码。如果用户输入的第二个参数是一个合法的值，还需要判断输入文件中的两个数是否合法。比如对于 8 进制来说：93 不是一个合法的值，因为 9 不是一个有效位。

程序的输出结果进制应当与输入数值的进制相同。

扩展 2：

在扩展 1 的基础上再增加一个参数，表示输出的进制。比如：

MyProm InputFile 8 3

表示计算两个 8 进制长整数相加，结果以 3 进制的形式输出。