**CS205 C/ C++ Program Design**

**Assignment 3**

**Name**: 巫晓, **SID**: 11912803

## Part 1. Life is short, show me the code.

为了方便多个版本的迭代，我将我的代码放在了GitHub上

https://github.com/XiaoLing12138/cppwork.git

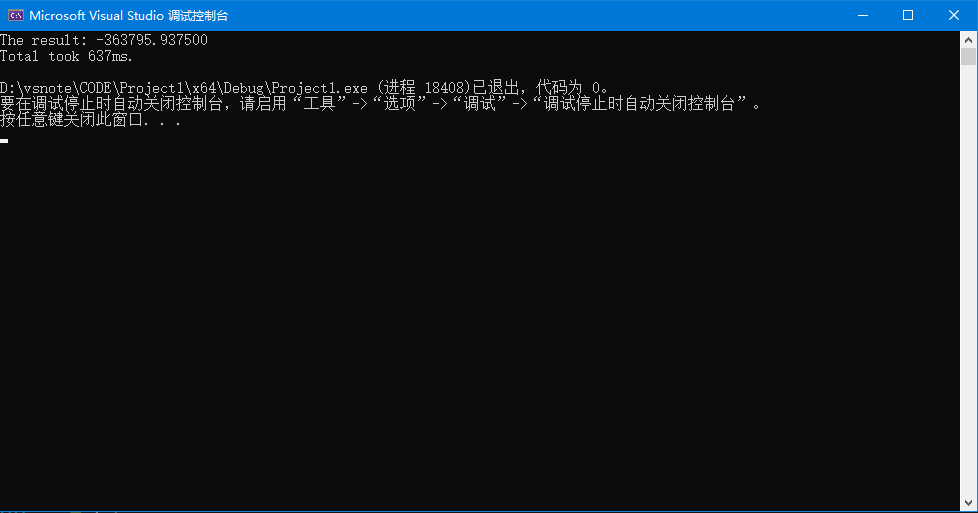
## Part 2. Result & Verification

作业中要求我们实现对两个向量的点积。实现这个功能非常简单，但是当数据量变大的时候，我们需要考虑很多问题，特别是当向量达到2亿维的时候。

比如说，如何输入输出，如何处理输入错误，如何加快运算。

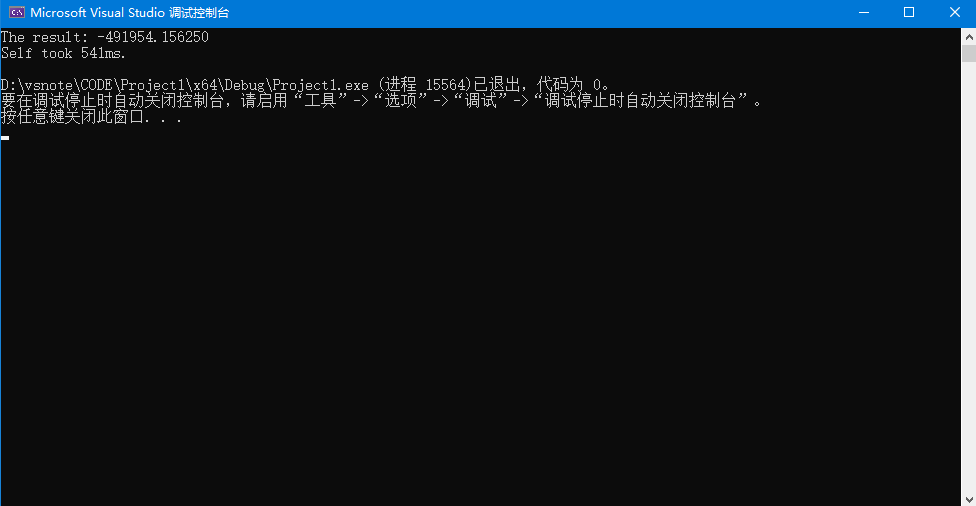
在第一个版本中，我实现了基本的功能，例如文本输入检测和计算功能。但是它所存在的问题是读入的缓慢，大数据文本读入使得程序需要运行非常久才能得到运算结果（估计要1.5h）。

于是我在第二个版本中使用了二进制文本读入，其加速效果十分显著，估计将IO时间缩短了50倍左右，此时也可以得到计算结果了。对于两个各自拥有2亿个元素的向量，其耗时约为650ms上下。

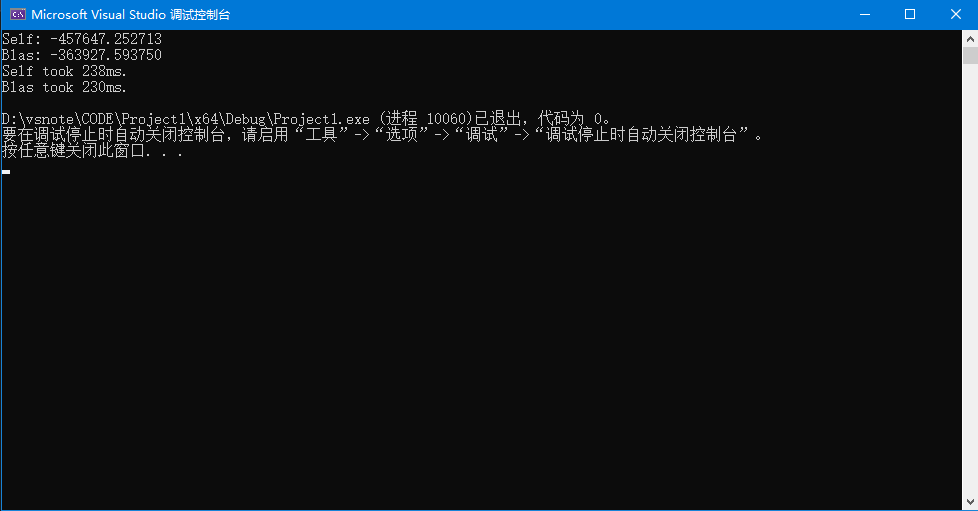


接下来我考虑的是计算加速的问题。

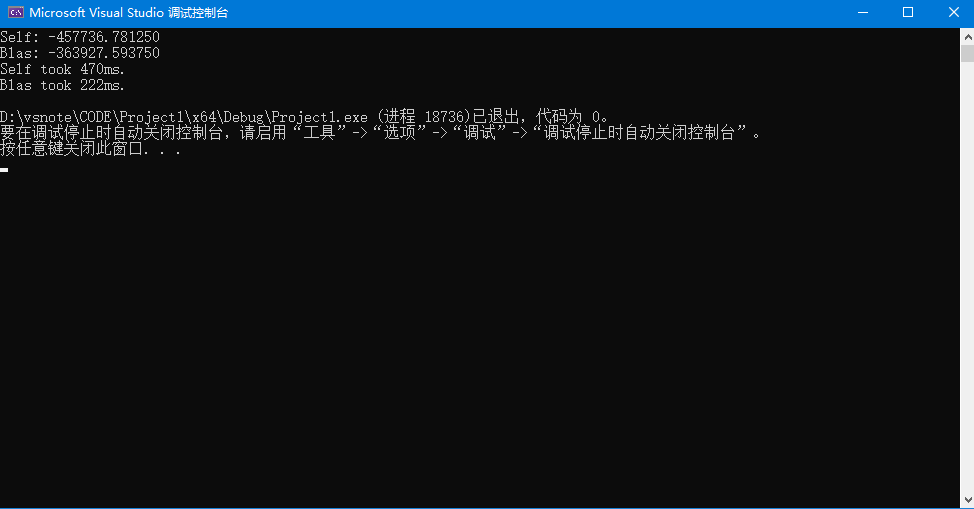
首先我想到的是使用多线程加速。在第三个版本中，我使用了八个线程同时进行运算。成功将运算时间缩短到了500ms上下。但是精度却大大下降。



然后，我找到了GitHub上的开源的项目，openBLAS。在第四个版本中，对比于openBLAS使用的浮点运算函数，我的计算耗时为它的2倍多。精度也不如openBLAS。



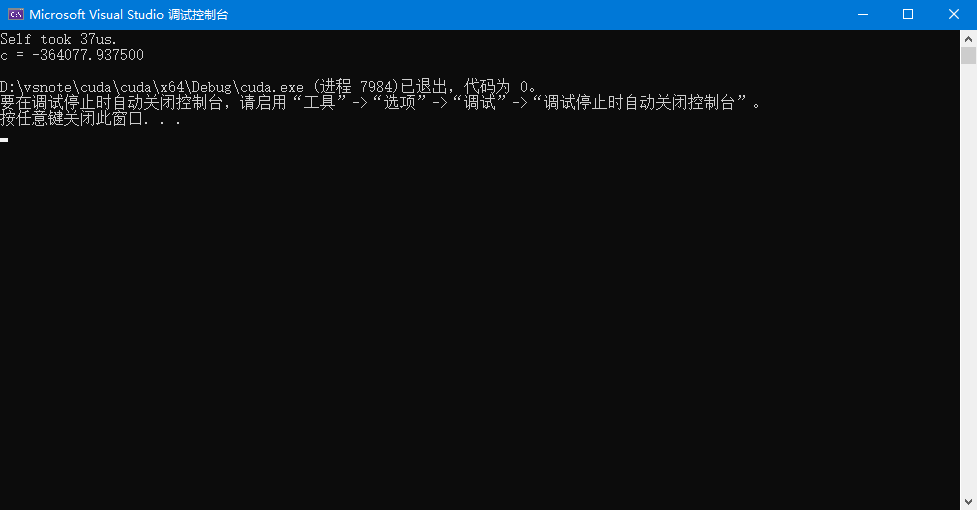
在计算过程中，我想到了答案精度问题。于是将存储结果的result变量类项改为double类型，结果起到了加速作用，可能是因为在运算中发生了浮点类型转换，而在我直接声明后，这一转换就去掉了，从而节省了大量时间。此时，我的代码与openBLAS计算所耗时间不分上下，均为220ms上下。虽然精度还是不如openBLAS。



最后，我在网上查到了cuda可以对计算进行加速，并尝试使用cuda。在第六个版本中，我实现的cuda代码耗时为1200ms上下，效果不尽如人意。精度也变差了。



我查阅了书籍之后，使用了《GPU高性能编程CUDA实战》上的代码，其加速效果十分好，2亿维向量相乘仅仅需要40us左右！而且结果与正确结果十分接近！



## Part 3. Difficulties & Solutions, or others

1. 难点：对于输入数据的验证

解决：将不合法的设置为0

1. 难点：文本输入输出太过费时

解决：使用二进制读入

1. 难点：单线程计算耗时长

解决：使用多线程进行运算

1. 难点：float可能存在精度缺失

解决：将结果设置为double类项，这样还能加速计算

1. 难点：运算能不能再快一些

解决：使用cuda对运算进行加速