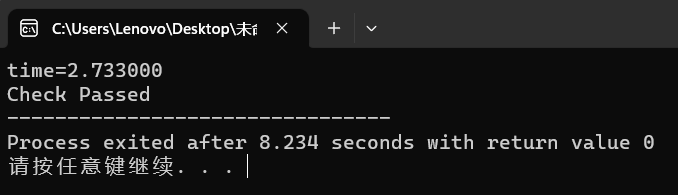
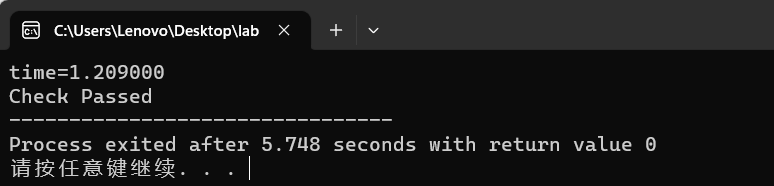
**实验二 Bonus：手写 SIMD 向量化实验报告**

1. **正确性和加速比**

基础版本：



SIMD 版本：

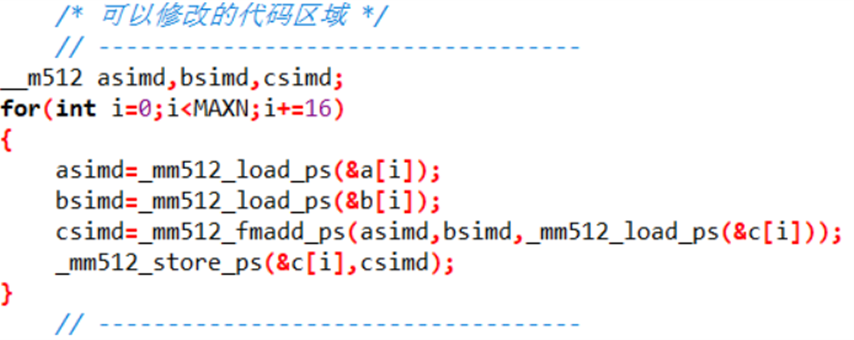
****

加速比=2.26

1. **代码及思路**

代码：

<SIMD.cpp>

****

思路：

1. \_\_m512:声明AVX-512向量寄存器变量，每个向量寄存器可以容纳16个单精度浮点数；
2. i+=16:设置循环步长为16，从0开始迭代，每次迭代处理16个单精度浮点数；
3. \_mm512\_load\_ps():该函数用于将数组中的元素存储到向量寄存器中，每次加载16个单精度浮点数；
4. \_\_m512 \_mm512\_fmadd\_ps(asimd,bsimd,\_mm512\_load\_ps(&c[i])):该函数用于将asimd和bsimd向量寄存器中的元素逐个相乘，然后将结果和数组c加载的16个单精度浮点数逐个相加，最后将结果存储在csimd向量寄存器中；
5. \_mm512\_store\_ps(&c[i],csimd):该函数用于将向量寄存器csimd中的16个单精度浮点数存储到数组c中；
6. 相较于SSE、AVX2、FMA指令集,通过测试可知AVX-512指令集并行计算能力更强，因为其一次可以处理16个单精度浮点数（前者是4或8），由于使用了AVX-512指令集，在编译选项中应加入-mavx512f。
7. **汇编代码对比**

<基础版本汇编.txt>

<SIMD版本汇编.txt>

明显可见优化后的汇编代码比优化前的汇编代码更长，表明优化后的代码包含更多的指令和优化技巧。通过观察可见mov、add、sub指令明显增多，这体现了优化后的代码使用了更多的寄存器来存储中间结果，而访问寄存器通常比访问内存更快，从而减少了内存访问的开销、提高了代码的执行效率。同时此类指令数量的增多也体现了优化后的代码使用了并行计算的优化技术以及SIMD向量化的算术优化例如执行乘法和累加操作的\_mm512\_fmadd\_ps()函数。