**姜庆彩-PB15051087-Project2**

1. **实验要求**

**实验格式:**

* a)实验需建立根文件夹，文件夹名称为：学号-project2，在根文件夹下需包括实验报告、和ex1、ex2两个子文件夹，每个子文件夹又分别包含3个子文件夹：
  + input文件夹：存放输入数据
  + source文件夹：源程序
  + output文件夹：输出数据
* b)input：
  + 实验1
  + 输入文件中每行一个随机数据，总行数大于等于31
  + 顺序读取N个数据，进行计算
  + example：计算N个矩阵的矩阵链乘问题，其随机数据的输入文件路径为

学号-project2/ex1/input/input.txt，顺序读取前N+1个数据进行计算。

* + 实验2
  + 输入文件中每行一个随机数据，总行数大于等于120
  + 先顺序读取N个数据，作为多项式A(x)的系数，再顺序读取N个数据，作为多项式B(x)的系数
  + example：计算两个多项式相乘的问题，其随机数据的输入文件路径为

学号-project2/ex2/input/input.txt，顺序读取前2N个数据进行计算

1. **实验环境**

编译环境：Xcode 11.0

机器内存：8G

时钟主频：2.9GHz

1. **实验过程**

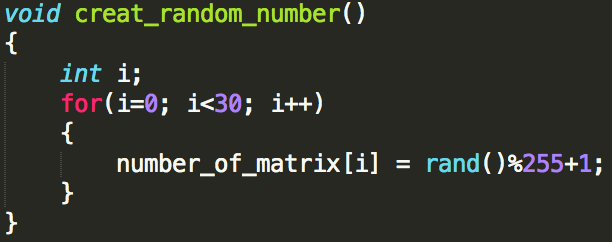
对于ex1矩阵链乘，首先调用create\_random\_numbers产生（0，256）之间的整数存于input.txt中，然后调用matrix\_chain\_order计算矩阵链乘的顺序，结果存与全局变量s[ ]与m[ ]中，最后调用print\_optimal\_parens输出链乘的结果并存于文件result.txt中。

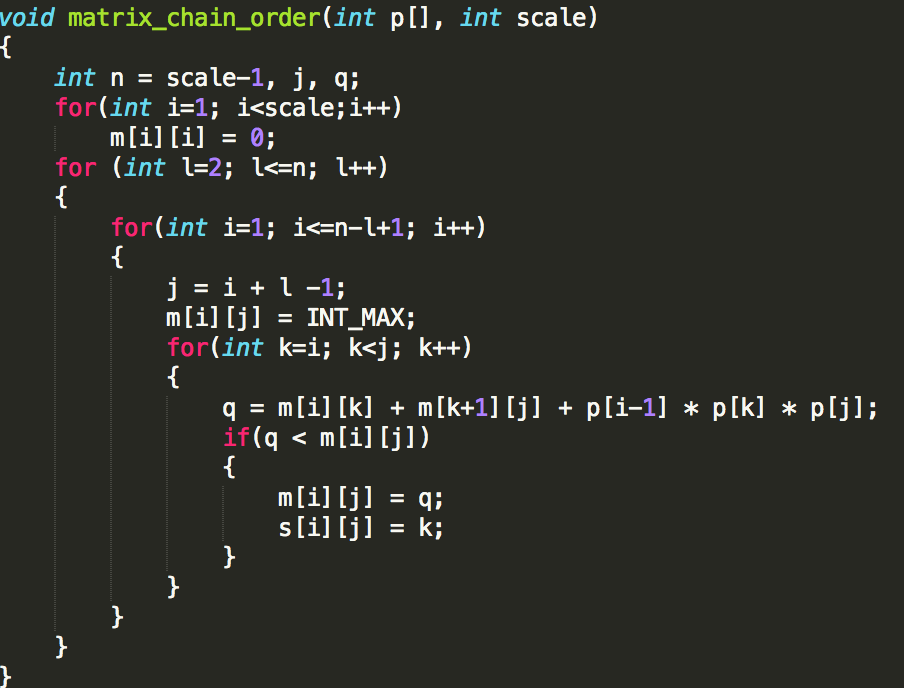
对于ex2 FFT算法，首先调用creat\_random\_number产生（-128，128）之间的随机有符号实数并存于input.txt中，normal\_solution函数表示普通乘法，FFT\_solution函数表示FFT算法，具体过程是将a[ ]与b[ ]高位补0后做FFT，结果相乘之后做逆FFT，再除以n，就得到了多项式相乘的结果。值得一提的是，recursive\_FFT(complex<double> arr[ ], int scale, int type)函数中type=-1时函数做的是逆FFT。

两实验都使用clock（）函数记录时间，不赘述。

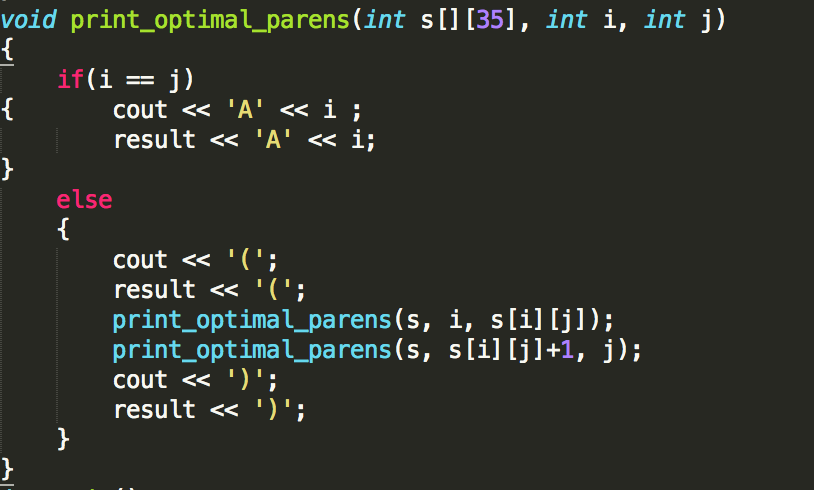
**4. 实验关键代码截图（结合文字说明）**

**ex1:**

****

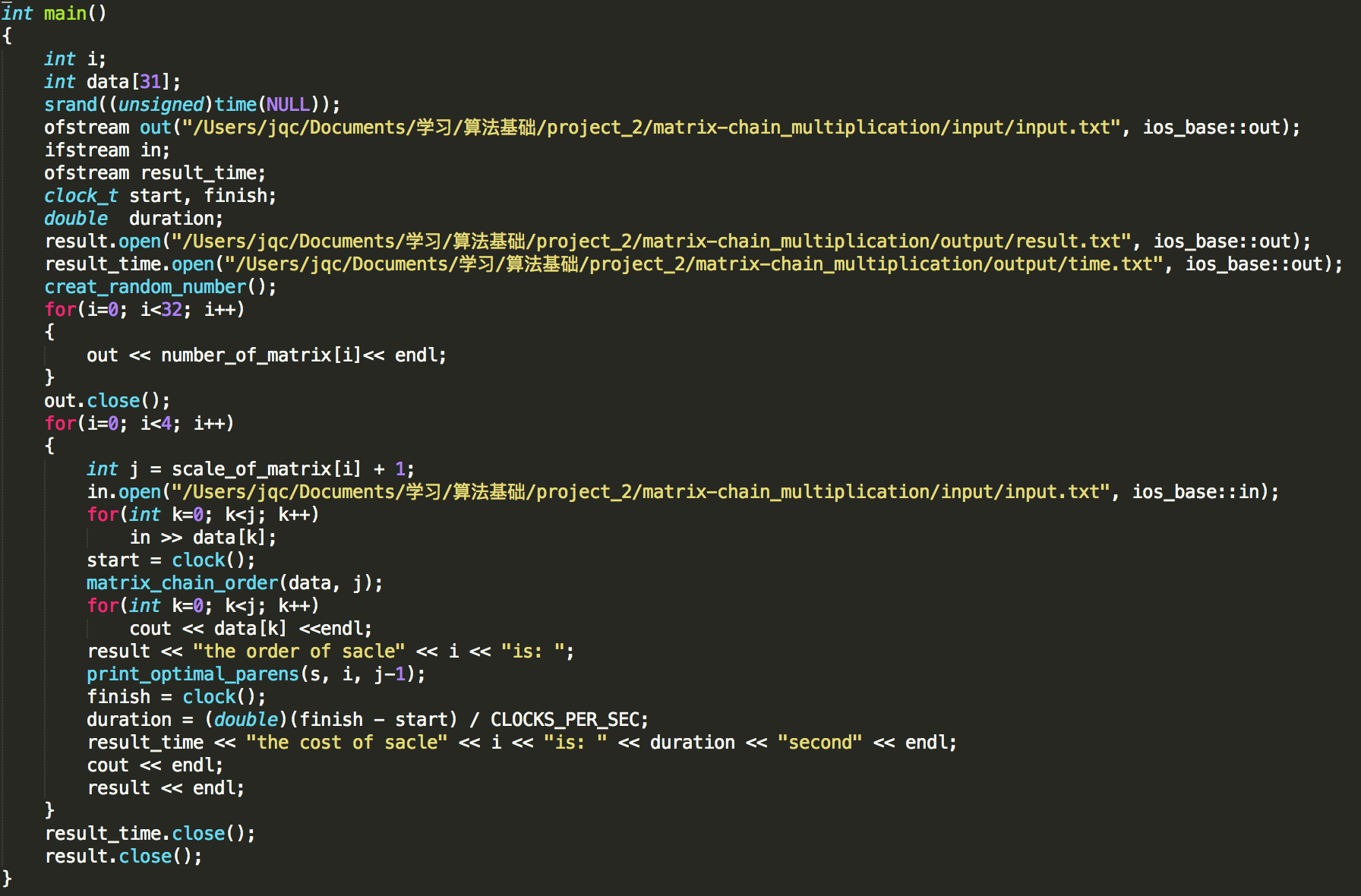


此函数与课本给出的完全一样

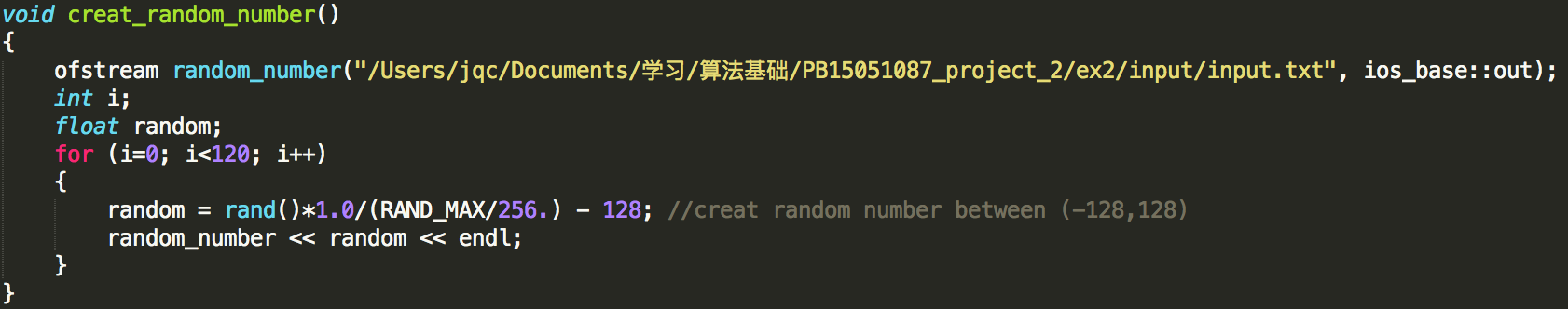


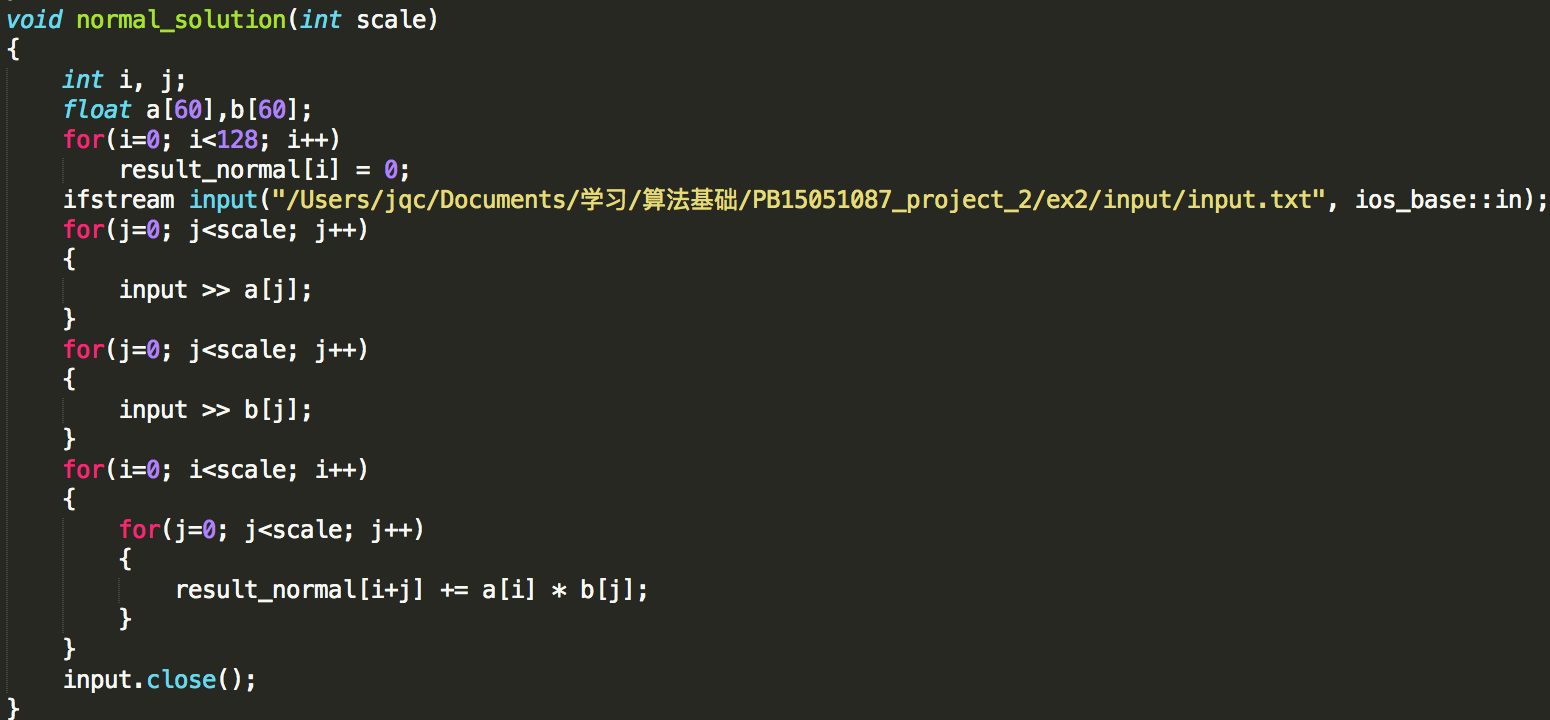
此函数同时将结果存在了input.txt中

main函数：

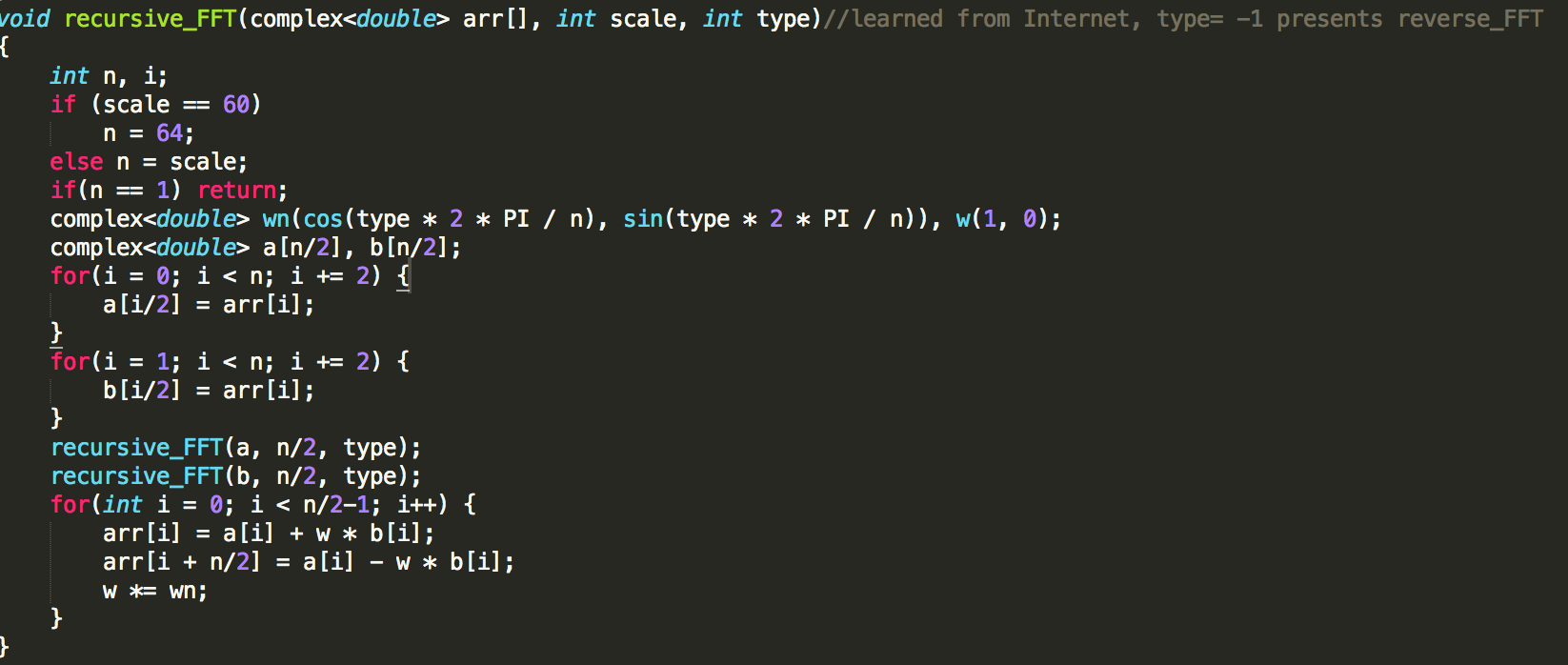


**ex2:**

****

****

两层循环实现普通乘法



type=-1时表示逆FFT

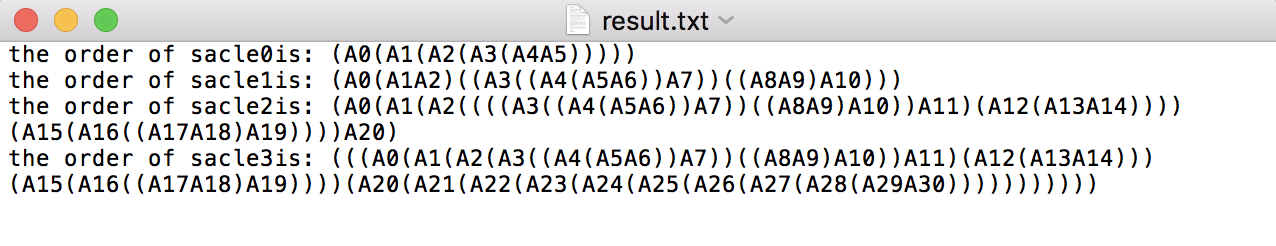


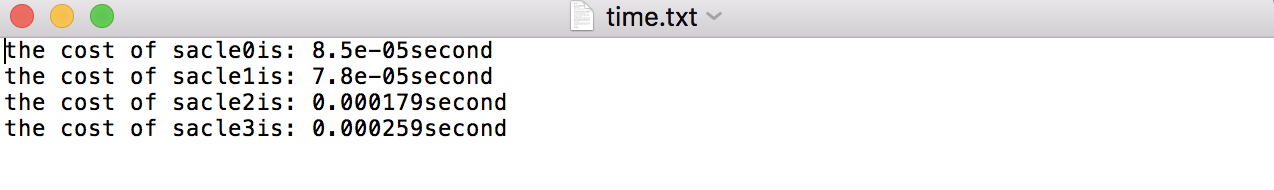
main函数：

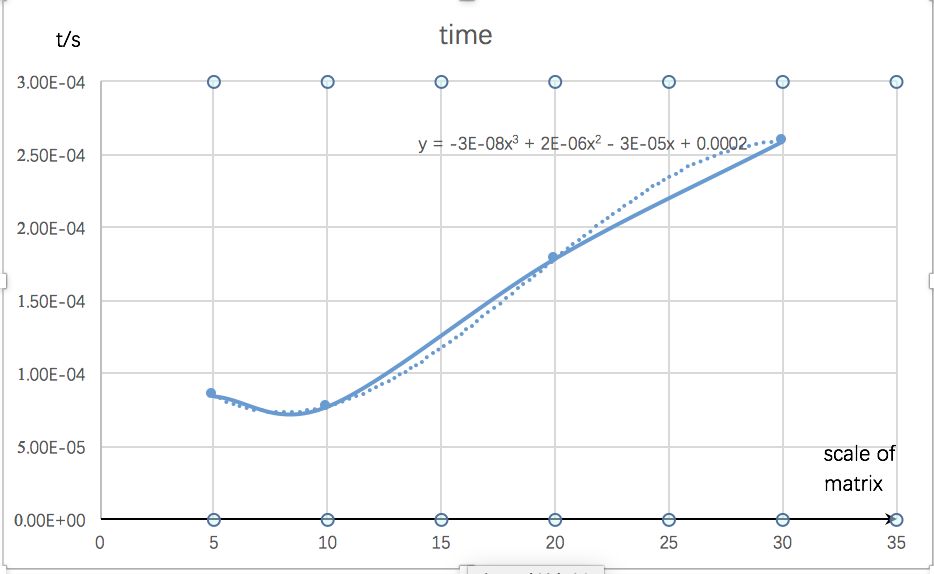


**5. 实验结果、分析（结合相关数据图表分析）**

**ex1:**

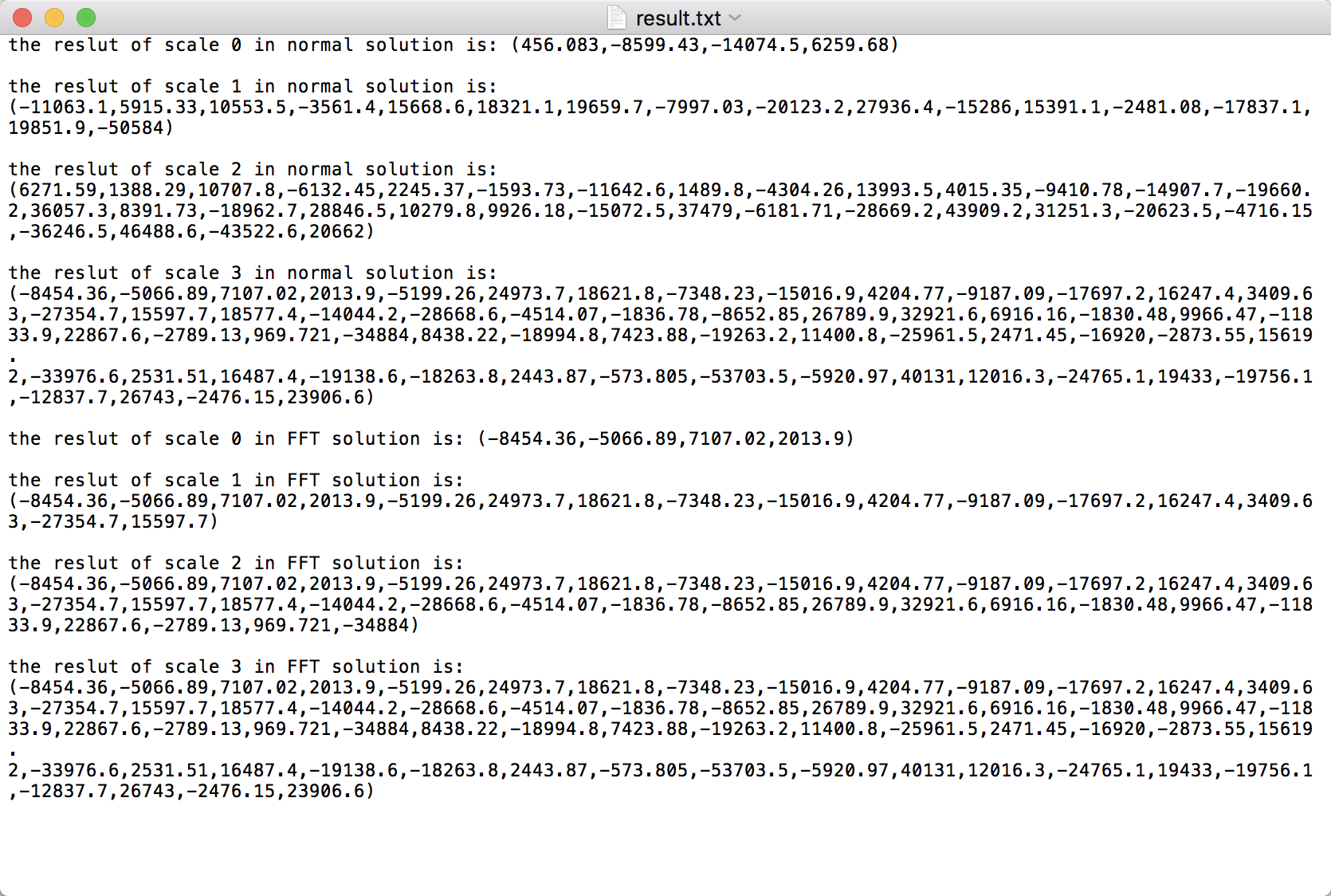
****

****

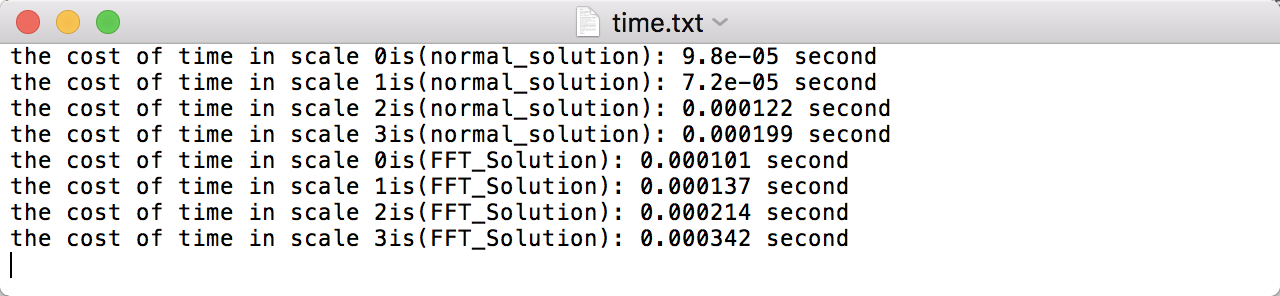
****

与课本中给出的O（n^3）的时间复杂度基本吻合。

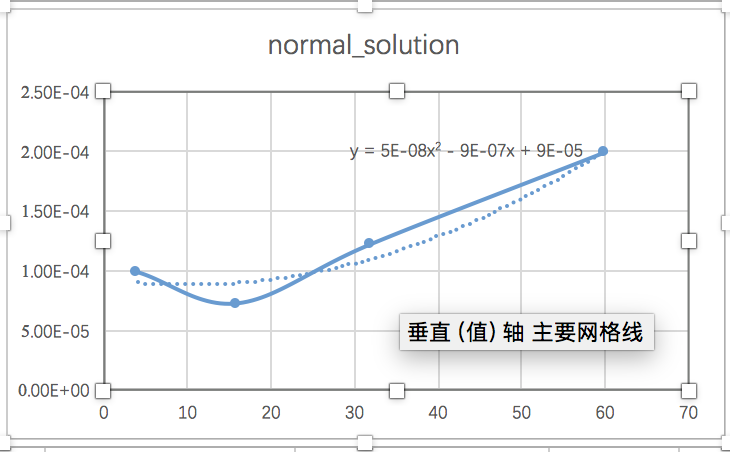
**ex2:**



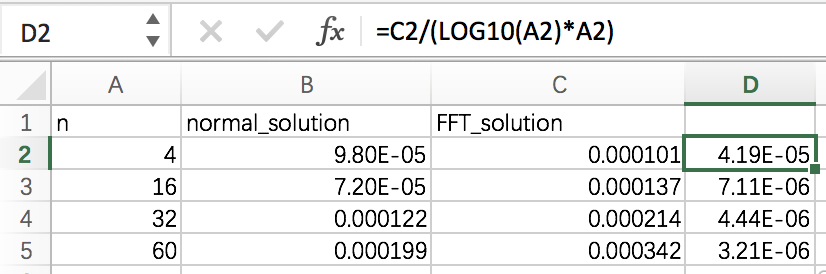
可以看出，两种计算方法得出的结果完全一样。



对时间进行分析：



与O（n^2）的时间复杂度吻合



可以看出，经过处理之后，D列的数据基本一致，这说明FFT\_solution的时间复杂度确实为O（nlogn）。

**6. 实验心得**

这次实验难度并不是很大，根据课本给出的算法基本可以比较顺利地完成，感谢老师与助教的指导！