实验报告

一.实验内容

采用归并排序，堆排序，快速排序，计数排序对输入数据进行排序，输出排序的结果和运行时间。

二.实验环境

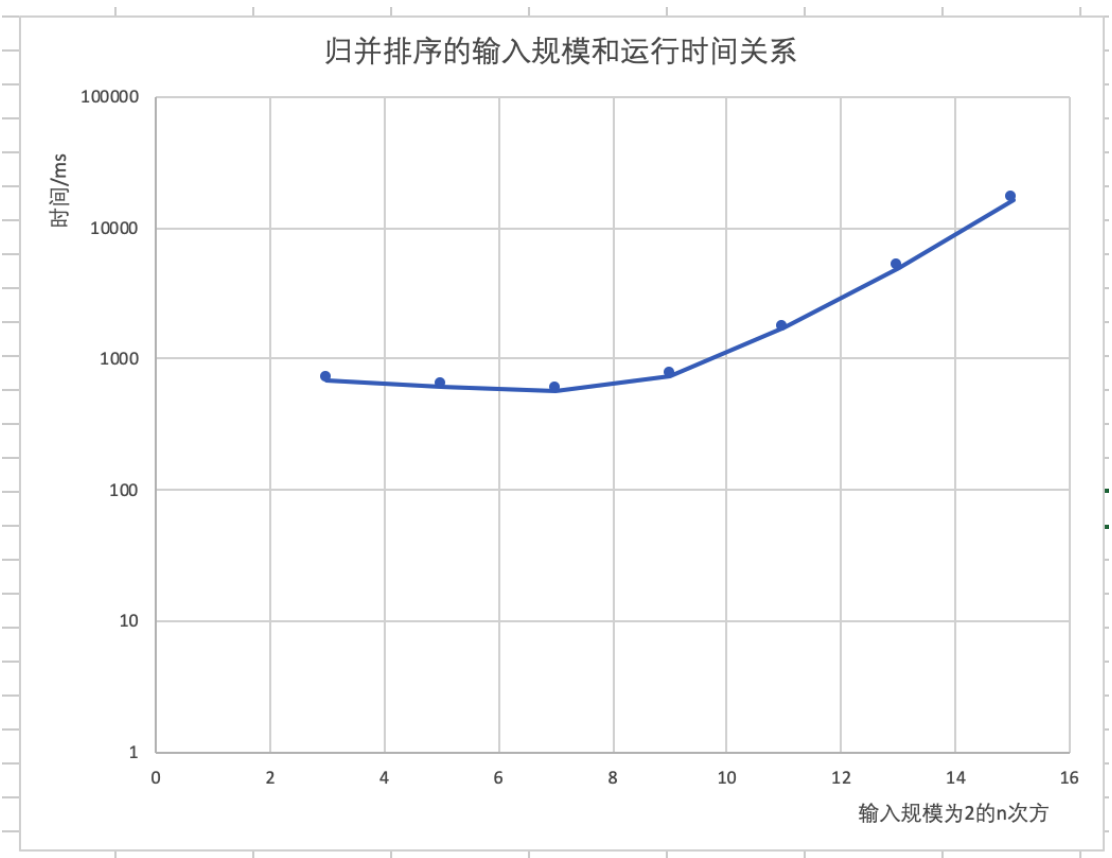
采用 C 语言实现本次实现，使用的工具为 MacOS 上的 Xcode。

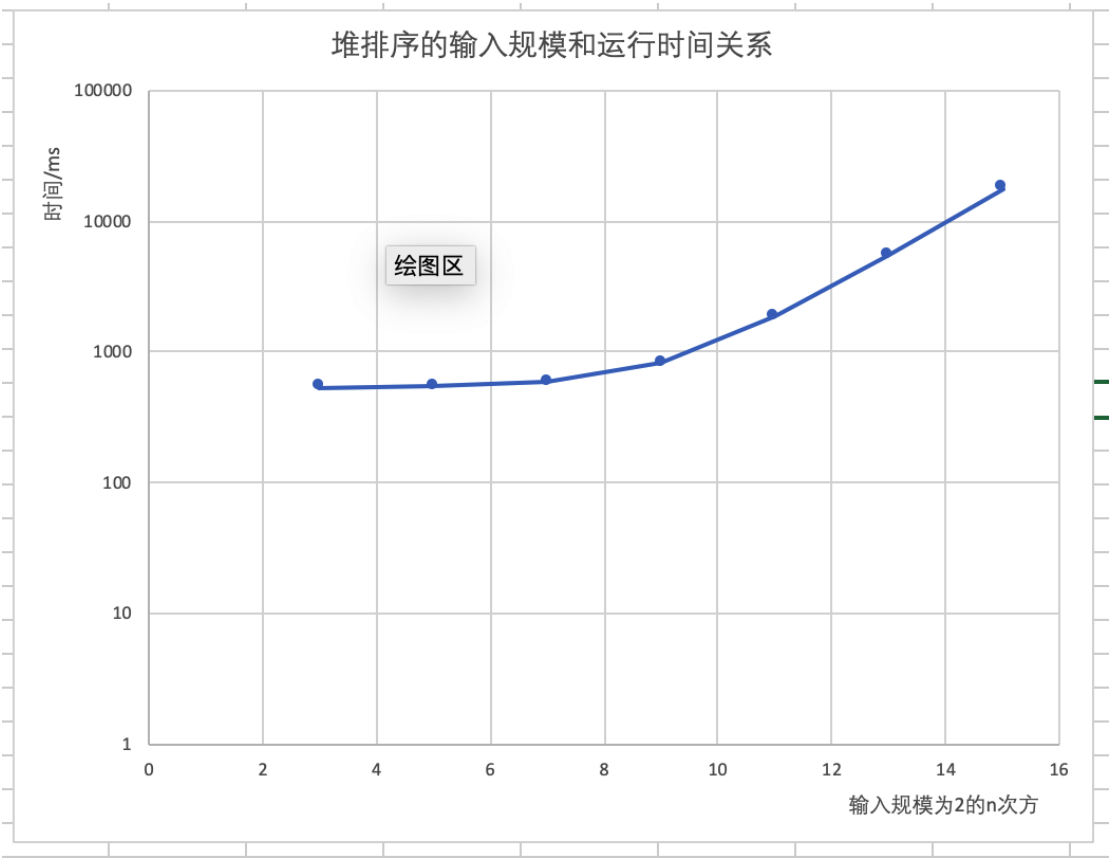
三.实验步骤

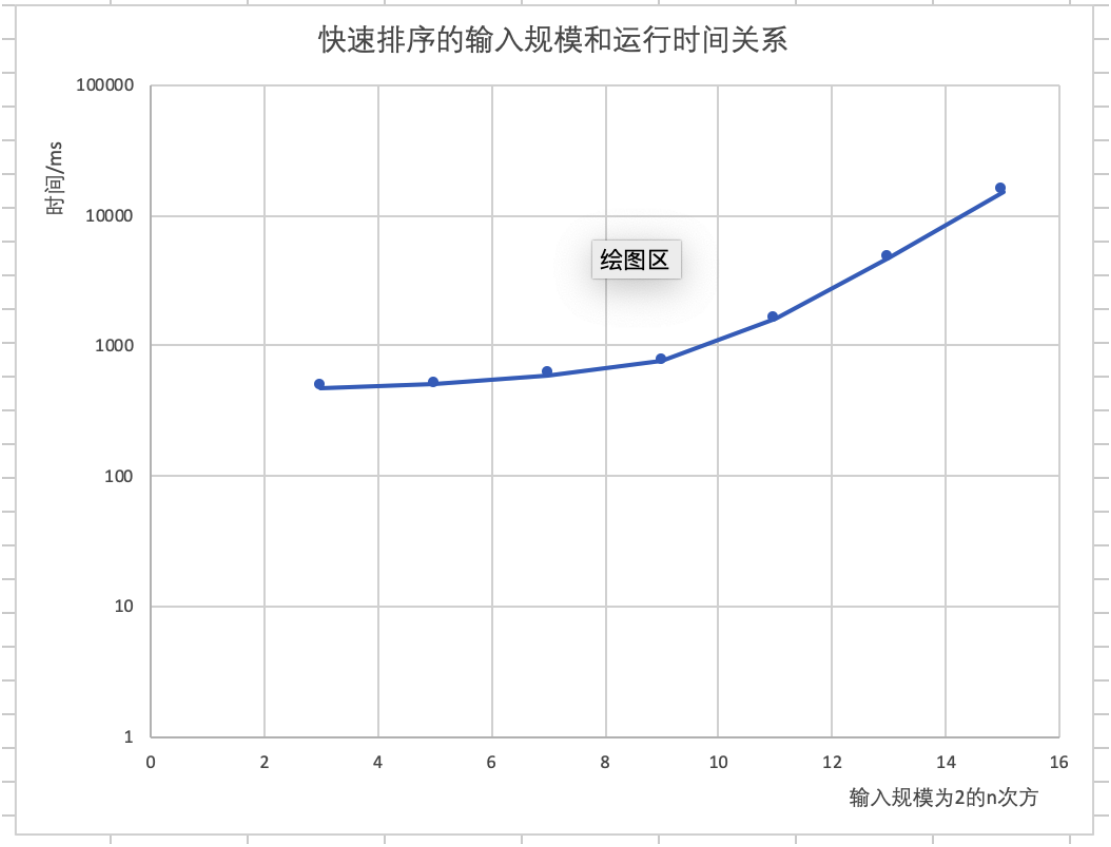
根据实验要求，创建了相应的文件夹，然后采用随机数的方法，用程序生成了 2^15 个随机数，存储在 input.txt 文件中。在 source 文件夹中，分别存储了四个程序的源代码，在 output 中用四个文件夹存储了 4 个程序的输出结果和运行时间。

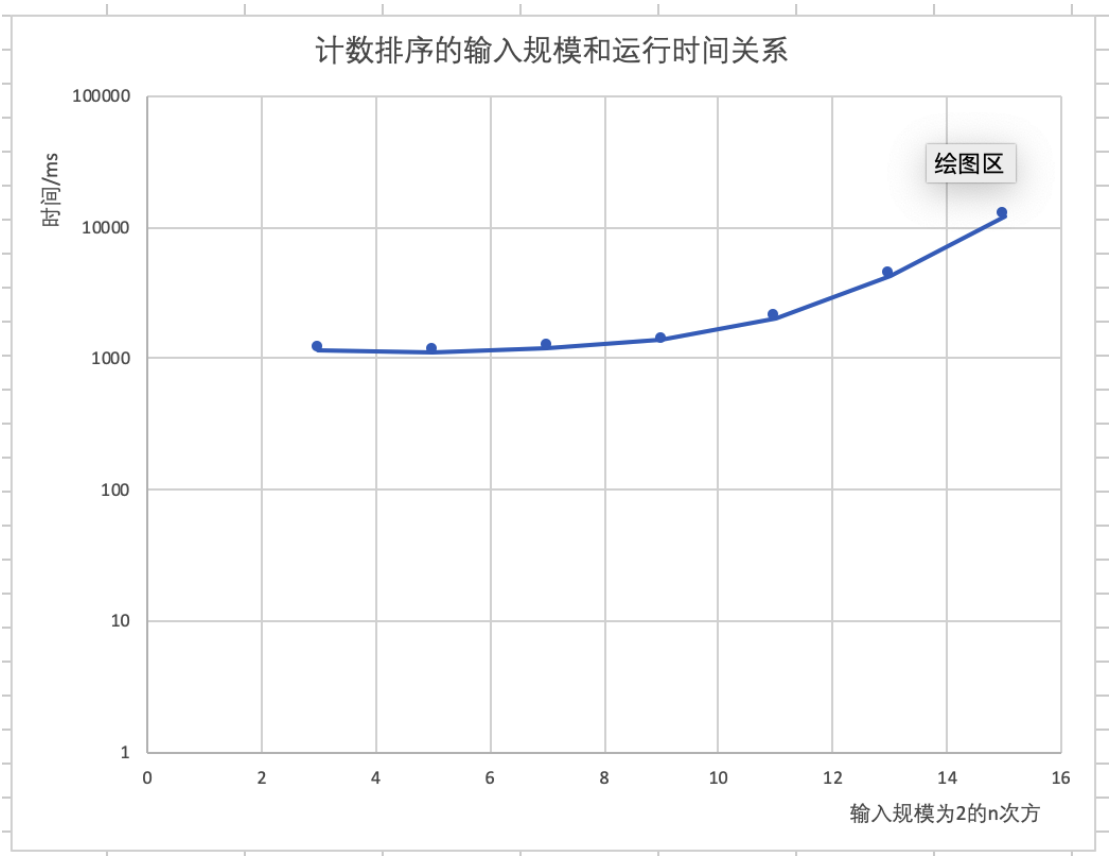
四.实验结果

四个排序的规模和运行时间的关系图：




堆排序的输入规模和运行时间关系
时间/ms
100000
10000
1000
100
10
1
绘图区
0
2
4
6
8
10
12
14
16
输入规模为2的n次方


快速排序的输入规模和运行时间关系
时间/ms
100000
10000
1000
100
10
1
绘图区
0
2
4
6
8
10
12
14
16
输入规模为2的n次方



从上面的四个曲线图可以看出，四种排序算法的渐进性能和书上的基本相同。

在不同的输入规模下，四种排序算法的性能也不一样。在输入规模较小的时候，快速排序和堆排序算法的运行时间相对较小，快速排序最小。在输入规模变大后，堆排序算法的时间变成最长，而计数排序的时间变为最小。

计数排序算法的时间变化波动相对最小，快速排序的性能一直都处于中上，堆排序不适合大规模数据处理，归并排序的性能相对一般。

五.代码说明

在这次实验中，采用了文件的输入输出格式，对于不同的输入规模，采用了在输入文件中取前相应规模个数作为输入数据。在每次程序运行时，需要从控制台输入规模的大小（即 2 的 n 次方中的 n），同时采用 sprintf 函数将输出文件名置为 name=result_n。

对于时间的计算，采用了 clock（）函数，用 begin 记录开始时间，用 end 记录结束时间，begin-end 即为程序运行所需时间，单位为 ms。同时运用 fopen 函数，将状态置为“a”，保证所有运行时间可以追加写进同一个文件中。

六.实验结果说明

实验结果在归并排序和计数排序的前几个时间计算上可能存在误差，因为显示的结果上，在规模小的时候，反而需要的时间变得更长。也可能是因为数据结

构的关系，在开始的几个数据中，因为过紧密或者过稀疏导致了处理时间变长，个人感觉还是时间统计的问题。