**实验设备：**

Laptop

**实验环境：**

macOS操作系统、Clion IDE

**实验内容：**

排序n个元素，元素为随机生成的1到65535之间的整数，n的取值为： 2^3，2^5，2^7，2^9，2^11， 2^13 ，2^15 ；

**实验要求：**

使用堆排序，快速排序， 归并排序，计数排序算法。A screenshot of a cell phone

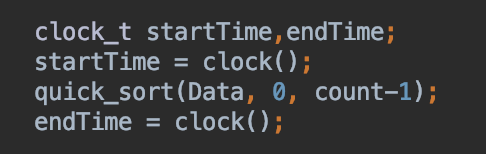
Description automatically generated

A screenshot of a cell phone

Description automatically generated

**实验方法：**

用clock函数记录程序运行排序算法时所花费的时间（以quick\_sort 举例）：

A black and silver text on a white background

Description automatically generated

（程序代码） （结果）

算法的实现主要来源于课本

**实验步骤：**

首先复习了一下C语言文件读写的知识以及如何产生足够多的随机数

再先将quick\_sort排序算法写好，之后文件读写部分代码基本无修改，主要是算法的部分进行改动

最后为每个算法测试运行时间并写入文件

A screenshot of a cell phone

Description automatically generated

**实验结果与分析：**

1. 经检查，排序算法均已完成且无错误
2. 算法在不同规模下的运行时间曲线

（图片的横坐标单位为数据个数，纵坐标为秒）

A screenshot of a cell phone

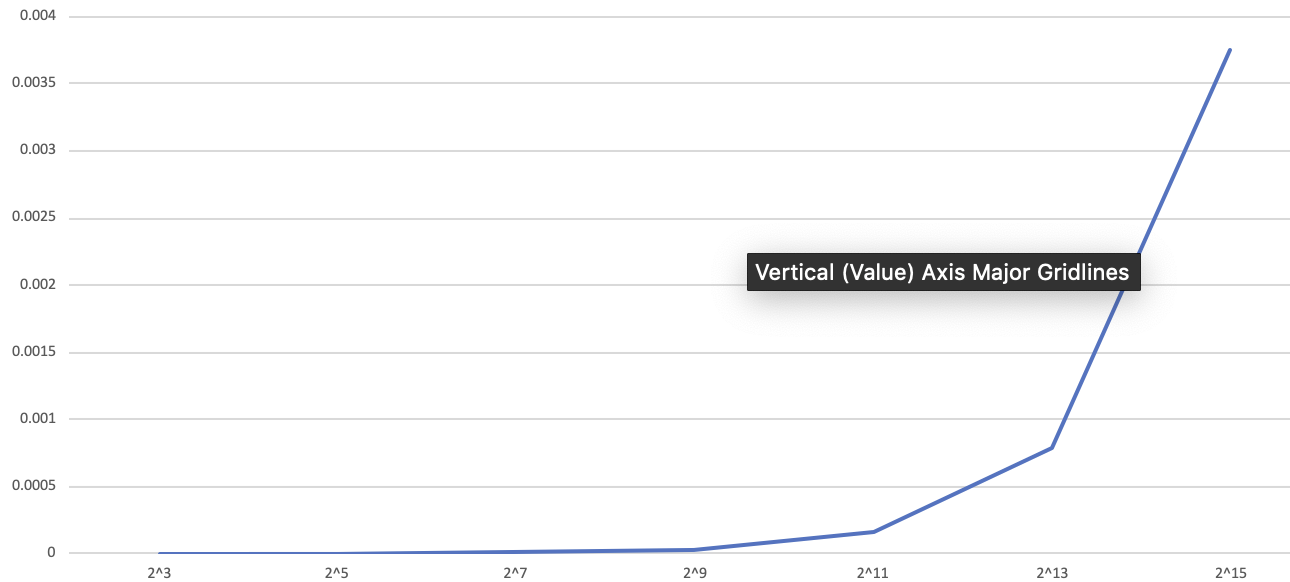
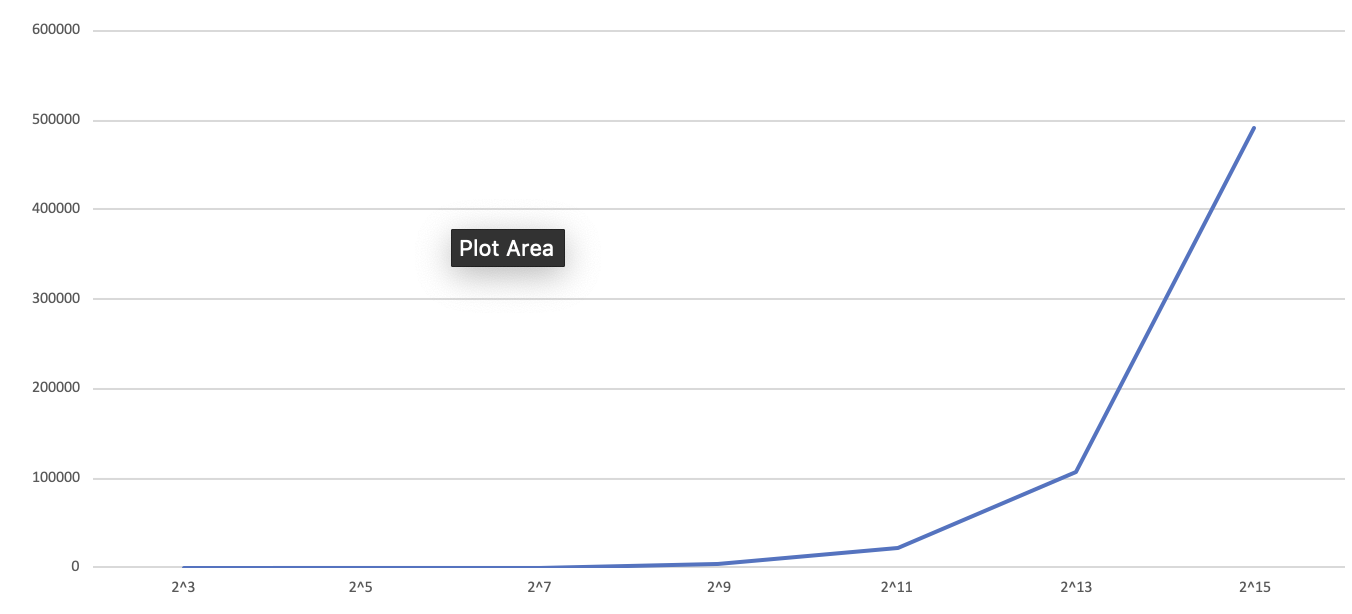
Description automatically generated

（上图最底下为quick\_sort 往上依次是merge\_sort、heap\_sort）（下图为count\_sort）

A screenshot of a cell phone

Description automatically generated

nlogn曲线（左）： quick\_sort曲线（右）：



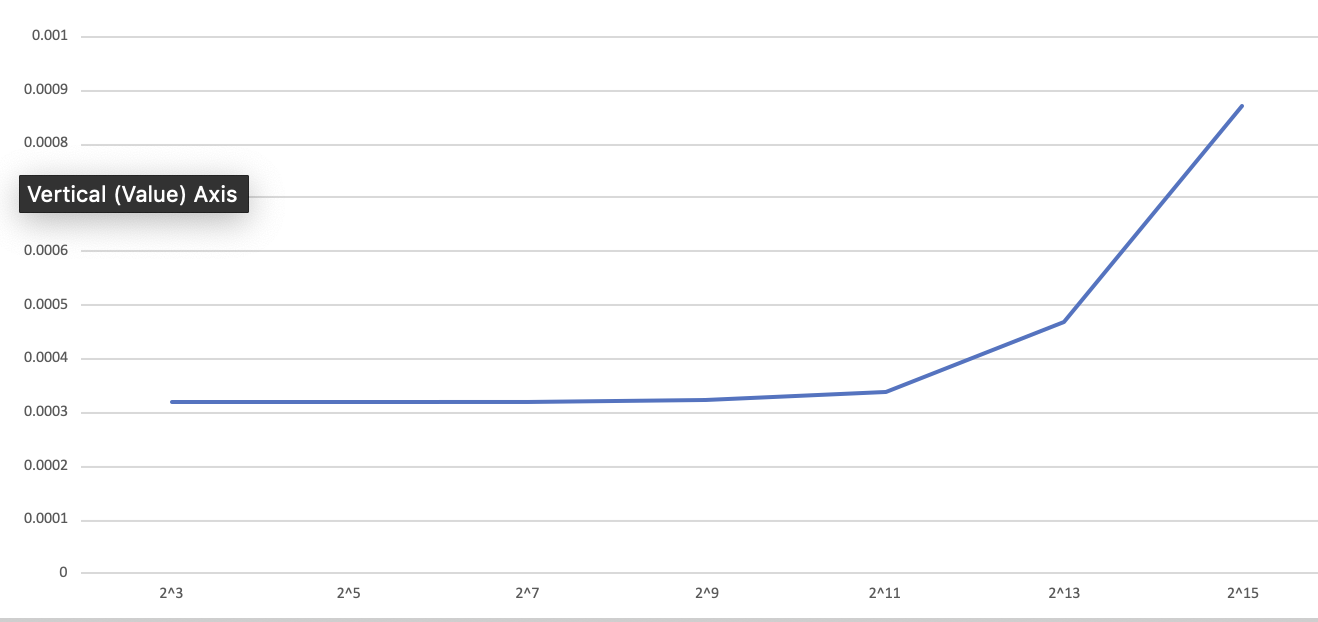
merge\_sort 曲线： heap\_sort曲线：

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Count\_sort曲线：



Quick\_sort、merge\_sort、heap\_sort的时间渐进复杂性均为nlgn，图像基本符合，而count\_sort的时间渐进复杂性应为O(n)，根据图像当N比较小的时候曲线和课本有出入，可能在N较低时测试时间有误差，又或者当N较小时常数项的影响比较大。

当N较低时quick\_sort、merge\_sort、heap\_sort算法的效率均比较高，当N较大时，count\_sort的效率比较高，因为不是基于比较的排序。