**实验报告八**

1. **实验内容**

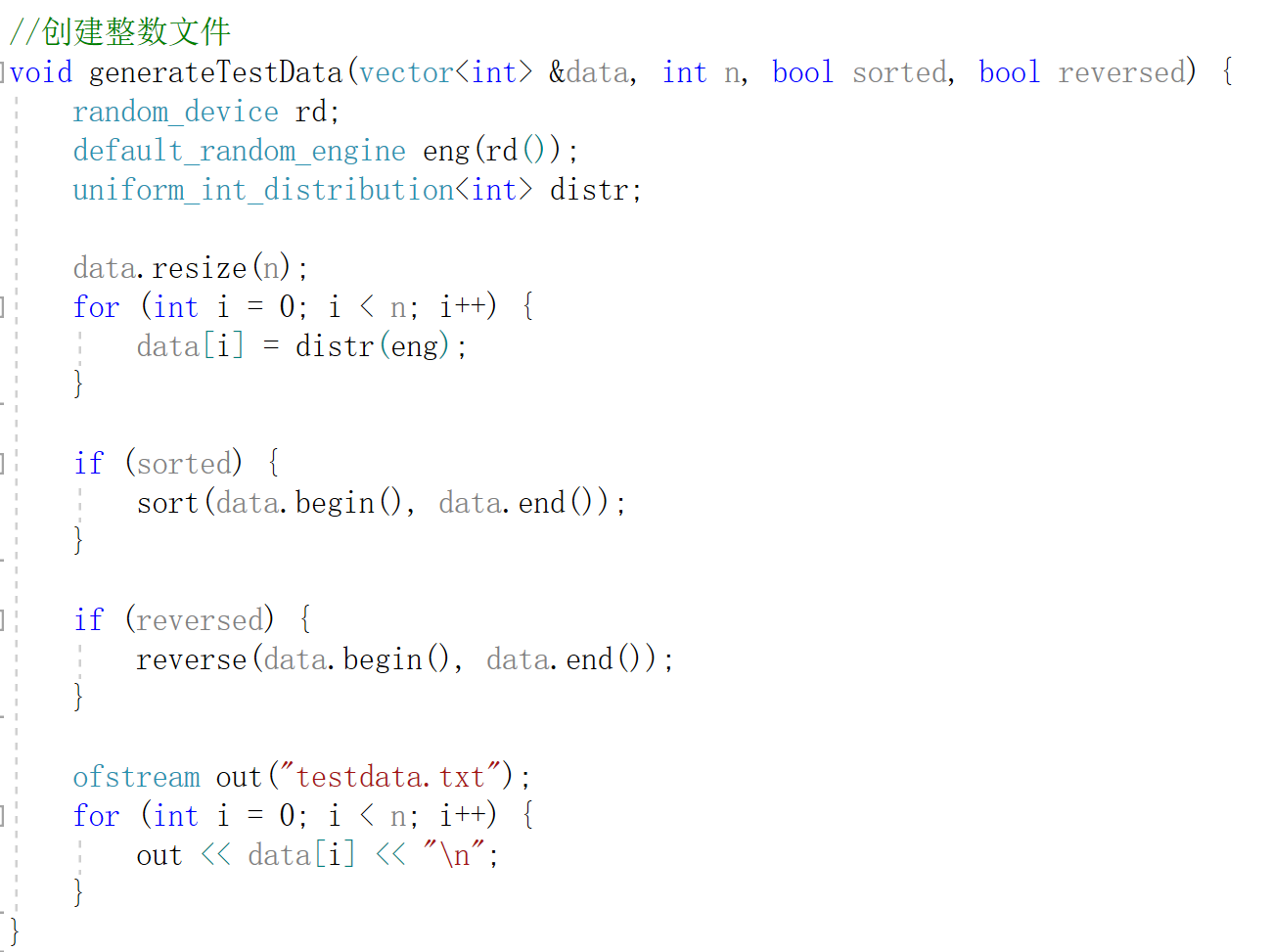
排序算法测试与比较

1. **数据结构设计与思路分析**
2. **本题所采用的数据结构知识：**

线性表+排序方法

1. **实验设计思路：**

1、创建用户可自定义数据量大小的数据集，采用了文件读写的操作。该函数的作用即为了控制变量，确保每次实验所采用的实验数据均相同。



1. 各种排序算法的设计

（1）插入排序（Insertion Sort）:

算法设计思路:

逐步构建有序序列，每次将一个元素插入到已经排好序的部分，直到整个序列有序。

从第二个元素开始，与前面已排序的元素比较，找到合适的位置插入。

时间复杂度:

最坏情况：O(n^2) - 当数组完全逆序时。

最好情况：O(n) - 当数组已经有序时。

平均情况：O(n^2)。

（2）选择排序（Selection Sort）:

算法设计思路:

每次选择未排序部分的最小元素，与未排序部分的第一个元素交换位置。

通过不断选择最小元素，逐步形成有序序列。

时间复杂度:

不论输入数据的初始排序如何，时间复杂度都是O(n^2)。

比较次数与交换次数都是固定的。

（3）希尔排序（Shell Sort）:

算法设计思路:

希尔排序是插入排序的改进版本，通过比较相隔一定间隔的元素，逐步减小间隔，最终得到有序序列。

希尔排序的核心在于选择不同的增量序列。

时间复杂度:

平均时间复杂度取决于选用的增量序列。

最坏情况：O(n^2)。

（4）归并排序（Merge Sort）:

算法设计思路:

利用分治法，将数组不断拆分直至单个元素，然后两两合并有序数组。

关键在于归并操作，将两个有序数组合并为一个有序数组。

时间复杂度:

始终是O(n log n)。

归并排序是稳定排序算法。

（5）快速排序（Quick Sort）:

算法设计思路:

通过选取一个基准元素，将数组划分为小于基准和大于基准的两部分。

递归地对子数组进行排序。

时间复杂度:

最坏情况：O(n^2) - 当数组已经有序时。

平均情况：O(n log n)。

（6）堆排序（Heap Sort）:

算法设计思路:

构建一个最大（最小）堆，使堆顶元素为最大（最小）值。

依次将堆顶元素与最后一个元素交换，然后重新调整堆。

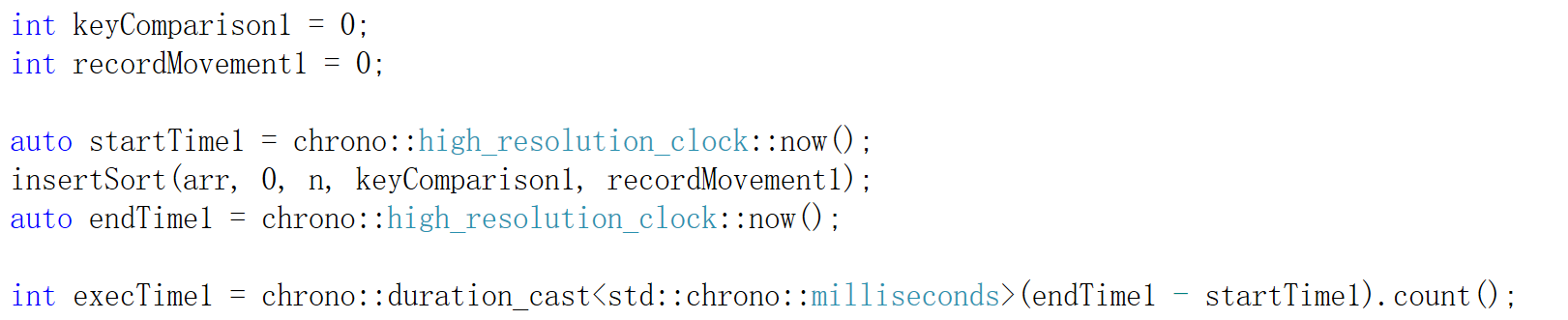
时间复杂度:

始终是O(n log n)。

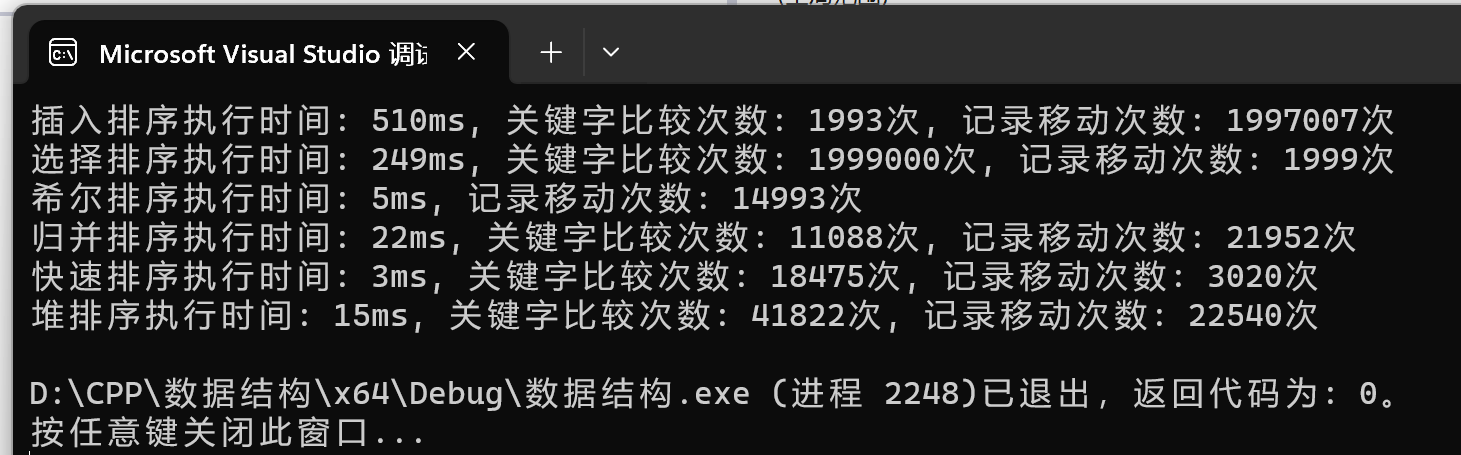
不稳定排序算法。

1. 如何计算运行时间：

在本次实验采用记录程序开始执行的当前时间以及程序执行结束时的当前时间然后做差得到最后结果，具体代码块如下：



1. **实验结果与分析**



由实验结果可见：希尔排序，快速排序，堆排序三种排序性能较好，具有更高的执行效率。

1. **实验小结**

通过本次实验，我对各种排序方法有了更深刻的理解。与此同时，我通过编写程序，直观地比较出各种排序算法的性能（耗时、关键字比较次数、移动次数）。除此之外，我还掌握了如何生成相同的大批量的实验随机数以控制变量。