项目说明文档

数据结构课程设计

——8种排序算法的比较案例

作 者 姓 名： 伊啸

学 号： 1951220

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

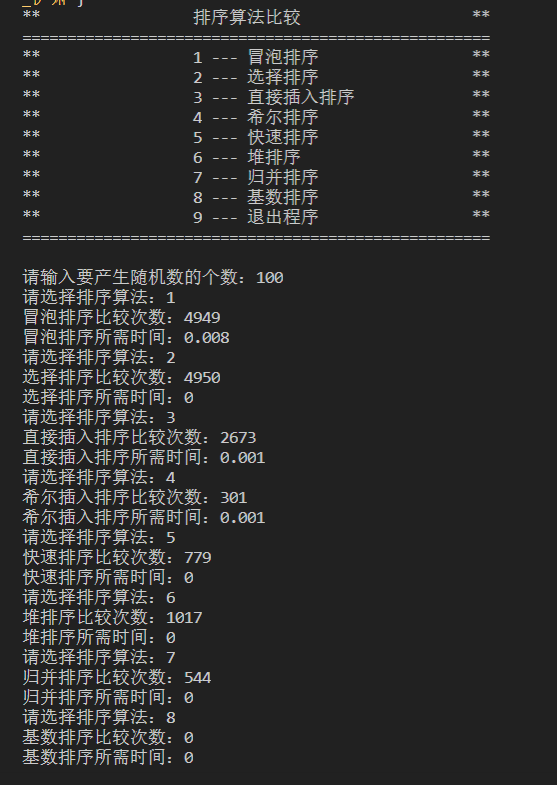
# 1 分析

## 1.1 项目简介

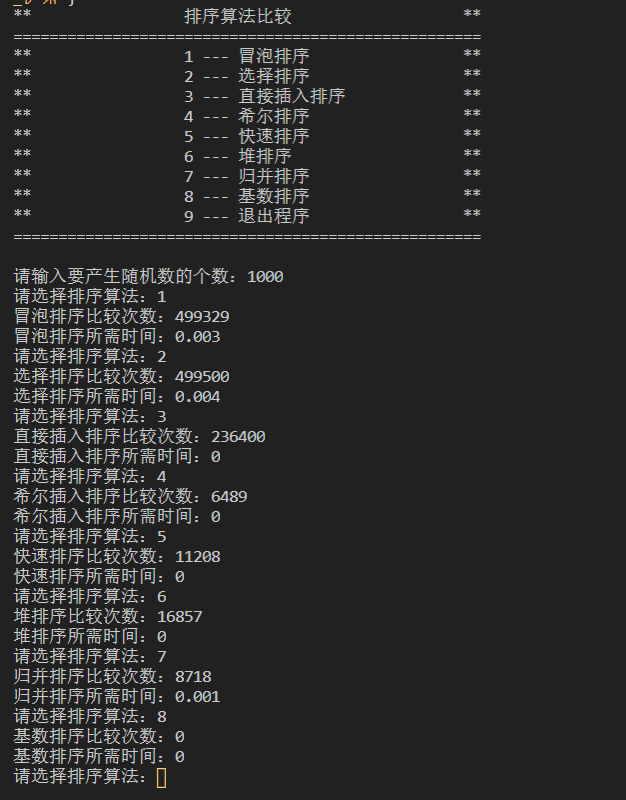
随机函数产生一百，一千，一万和十万个随机数，用快速排序，直接插入排序，冒泡排序，选择排序的排序方法排序，并统计每种排序所花费的排序时间和交换次数。其中，随机数的个数由用户定义，系统产生随机数。并且显示他们的比较次数。

# 2 实现

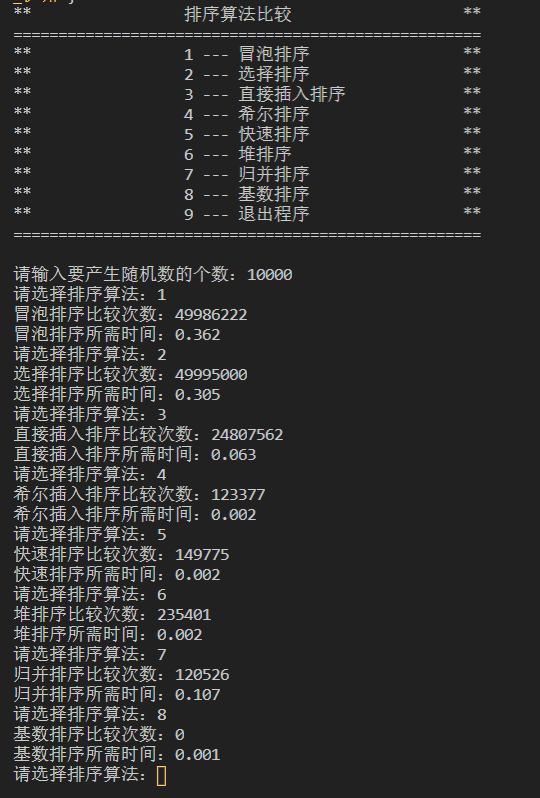
随机数为100：



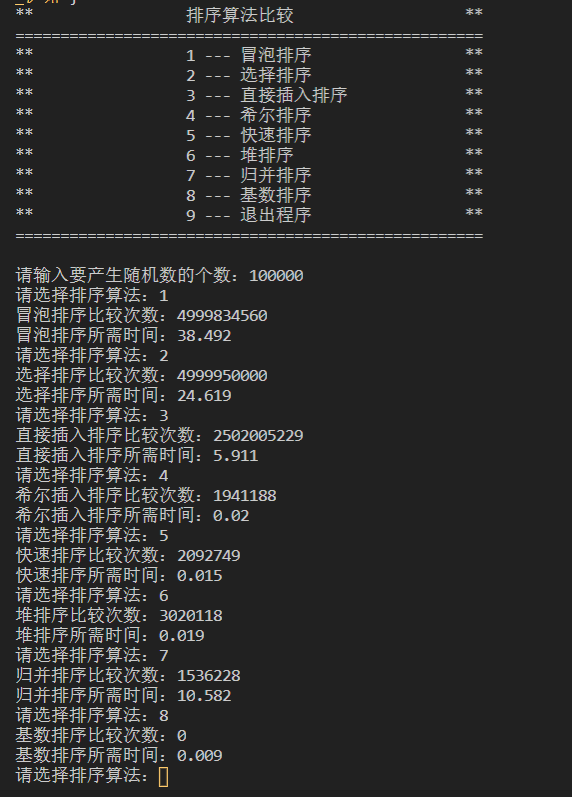
随机数为1000：



随机数为10000：



随机数为100000：



**冒泡排序**：排序时间和交换次数最长，但空间开销小，时间复杂度为O（n²）。每进行一次循环都需要比较所有元素，最好的情况下可能一次也不换，最差情况下每一次比较都需要交换，一般情况下大约需要n²/2次比较和交换操作。但在整个过程中只有交换和循环的时候需要一个额外的空间，因此冒泡排序的空间开销相对来说比较小。此外，冒泡排序没有跨元素交换，因此是稳定的排序方法。

**选择排序**：选择排序的排序码比较次数与元素的初始排列无关。第i趟选择具有最小排序码元素所需的比较次数总是n-i-1次,此处假定整个待排序元素序列有n个元素。因此,总的排序码比较次数约为n²/2，元素的移动次数与元素序列的初始排列有关。当这组元素的初始状态是按其排序码从小到大有序的时候,元素的移动次数为0,达到最少;而最坏情况是每一趟都要进行交换,总的元素移动次数为3(n-1)。尽管如此,相比于其他排序算法,待排序元素序列的有序性对于选择排序的运行时间的影响不大。因为从未排序部分选择最小元素的每步操作过程,没有对下一步要找的最小项的位置给出相关信息。因此,对于已经排好序的序列或者各元素排序码值完全相等的序列,选择排序所花的时间与对随机排列的元素序列所花的时间基本相同。因此选择排序比较简单并且执行时间比较固定。而且它对一类重要的元素序列具有较好的效率,这就是元素规模很大,而排序码却比较小的序列。因为对这种序列进行排序,移动操作所花费的时间要比比较操作的时间大得多,而其他算法移动操作的次数都要比选择排序来得多。选择排序存在跨元素的交换，是一种不稳定的排序方法。

**直接插入排序：**假设待排序的元素个数为n,则该算法的主程序执行n-1趟。因为排序码的比较次数和元素移动次数与元素排序码的初始排列有关,所以在最好情况下,即在排序前元素已经按排序码的大小从小到大排好序了,每趟只需与前面的有序元素序列的最后一个元素的排序码比较1次,总的排序码比较次数为n-1,元素移动次数为0。而在最差情况下,即第i趟时第i个元素必须与前面i个元素都做排序码比较,并且每做1次比较就要做1次数据移动,则总的排序码比较次数和元素移动次数分别为都约为n²/2。

直接插入排序的运行时间和待排序元素的原始排列顺序密切相关，若待排序元素序列中出现各种可能排列的概率相同,则可取上述最好情况和最差情况的平均情况。在平均情况下的排序码比较次数和元素移动次数约为n²/4,因此,直接插入排序的时间复杂度为O(n²),直接插入排序是一种稳定的排序方法。因此选择排序的时间复杂度为O（n²），但在整个过程中只有交换的时候需要一个额外的空间，因此直接插入排序的空间开销相对来说比较小。此外，直接插入排序没有跨元素交换，因此是稳定的排序方法。

**希尔排序：**由结果可知，希尔排序交换次数和所需时间相比前几个明显较少，但会需要稍微大一点的空间开销。即使对于规模较大的序列(n<=1000) ,希尔排序都具有很高的效率。并且希尔排序算法的代码简单,容易执行,所以很多排序应用程序都选用了希尔排序算法。希尔排序有跨元素的交换，是一种不稳定的排序算法。

**快速排序**：从结果来看，快速排序所需时间以及交换次数是较快的，大概和希尔排序差不多。如果每次划分对一个元素定位后,该元素的左侧子序列与右侧子序列的长度相同,则下一步将是对两个长度减半的子序列进行排序,这是最理想的情况。在n个元素的序列中,快速排序的时间复杂度为O(nlogn)级别的。此外,实验结果也表明:就平均计算时间而言,相对来说，快速排序是我们所讨论的所有内部排序方法中最好的一个。由于快速排序是递归的,需要有一个栈存放每层递归调用时的指针和参数,最大递归调用层次数与递归树的深度一致,理想情况为[log2(n+1)],因此,要求存储开销为O(log2n),但是,我们每次都选用序列的第一个元素作为比较的基准元素。这样的选择简单但不理想。在最坏的情况,即待排序元素序列已经按其排序码从小到大排好序的情况下,其递归树成为单支树。每次划分只得到一个比上一次少一个元素的子序列。这样,必须经过n-1趟才能把所有元素定位,而且第1趟需要经过n-1次个比较才能找到第1个元素的安放位置,第2趟需要经过n-2次排序码比较才能找到第2个元素的安放位置,…,总的排序码比较次数将达到n-1)其排序速度退化到简单排序的水平,比直接插入排序还慢。占用附加存储(即栈)将达到O(n)。

**堆排序：**堆排序的时间复杂度为O(nlog2n),该算法的空间复杂性为0(1)。堆排序是一个不稳定的排序方法。平均时间上，堆排序的时间常数比快排要大一些，因此通常会慢一些，但是堆排序最差时间也是O(nlogn)的，这点比快排好。比较次数和所需时间都是和快排差不多的，但相比冒泡排序和选择排序，堆排序要快的多，效率也高的多，但堆排序是不稳定的算法。

**归并排序：**由结果可知，归并排序所需时间是介于快排、堆排序和冒泡排序之间的，运行的时间复杂度为O(nlog2n),空间代价为O(n)。相比堆排序，归并排序占用的辅助内存更大，但它是稳定的排序算法。此外，在进行外排序时多用归并排序，不仅仅是二路归并，多路归并也可能出现。

**基数排序：**由结果可知，基数排序是所有排序中最快的。但在我看来，其实是因为随机数的数量和随机数的范围相同导致的，在这种情况下，基数排序只需要遍历n个元素和m个桶的数量就能完成排序，因此基数排序在这样的随机数机制下时间复杂度达到了O（n），所以其所需时间是最短的。基数排序的空间开销也不大，需要额外的桶来辅助排序。基数排序没有跨元素的交换，因此基数排序也是稳定的。