**第十章** **内部排序1**

1.对关键字序列（21,19,37,5,2），经直接插入排序法由小到大排序，第一趟后所得结果为【**(19，21，37，5，2)**】。

2.对关键字序列（21,19,37,5,2），经冒泡排序法由小到大排序，第一趟后所得结果为【**(19，21，5，2，37)**】。

3.对关键字序列（149,138,165,197,176,113,127），采用基数排序的第一趟之后所得结果为【**(113，165，176，197，127，138，149)**】。

4.下列各项键值【**(5，23，68，16，94)**】序列不是堆的。

5.假设一组待排序的关键字序列为（24,62,36,19），要求从小到大进行排序，【**(24，62，19，36) (19，24，36，62)**】是归并排序的过程。

6.在第一趟排序之后，不能确保将数据表中某一个元素放在其最终位置上的排序算法是（**归并排序**）。

7.对于下列排序，（**直接插入排序**）的时间效率与关键字初始序列有直接关系。

8.对于下列排序，（**归并排序**）的最坏时间复杂度是O(n㏒n)。

9.假设两个有序表长度分别为n和m，将其归并成一个有序表最少需要（**min{n,m}**）次关键字之间的比较。

10.对于下列排序，（**归并排序**）需要额外辅助存储空间达到O(n)。

11.对于关键字序列(49,38,65,97,76,13,27,49)，完成创建的大根堆是【**(97,76,65,49,49,13,27,38)**】。

12.对关键字序列（30，26，18，16，5，66），进行2遍（**选择**）排序后得到序列（**5，16，18，26，30，66**）。

13.在下列排序算法中，（**插入**）排序算法可能出现如下情况：在最后一趟排序之前，所有元素均不在其最终的位置上。

14.在下列排序方法中，（**快速**）排序方法的平均时间复杂度不是O（ ）。

15.假设两个有序表长度分别为n和m，将其归并成一个有序表最多需要（**n+m-1**）次关键字之间的比较。

16.下列排序算法中，（**冒泡**）排序算法是稳定的。

17.假设待排序的表长为n，那么下列排序算法中，（**归并**）排序算法需要O(n)的辅助空间。

18.假设待排序的表长为n，那么快速排序算法需要【**O(㏒n)**】的辅助空间。

19.在下列排序算法中，（**表插入**）排序算法可以避免在排序过程中移动数据元素。

20.‏假设待排序的表长为n，那么创建堆需要时间复杂度为【**O(n)**】。

21.在下列排序算法中，在待排序序列为有序的情况下，（**快速排序**）的时间复杂度是O( )，其中n为待排序序列的数据元素个数。

22.下列四种排序中，（**快速排序**）的辅助空间复杂度是最高的。

23.设哈希表为HT[0..16]，哈希函数H(key)=key%13，采用线性探测开放地址法处理冲突，且HT中已有关键字为11、28、47和18这4个数据元素，现插入关键字为24的数据元素，其实际存储的地址是（**12**）。

24.‌对顺序表中的n个记录进行直接插入排序，在最好情况下需要比较（**n-1**）次关键字。

25.‍排序算法的稳定性是指（**经过排序后，能使原来关键字值相同的数据保持原有顺序中的相对位置不变**）。

26.简单插入排序算法是不稳定的（**错误**）。

27.待排序记录关键字出现有序的初始排列时，快速排序的时间复杂性达到最坏情况（**正确**）。

28.相对于简单插入排序而言，半插入排序减少了关键字比较和移动的次数（**错误**）。

29.对顺序表中的n个记录进行直接插入排序，在初始关键字序列为逆序的情况下，需要关键字比较的次数最少（**错误**）。

30.对顺序表中的n个记录进行简单选择排序，至多需要关键字交换n-1次（**正确**）。

31.堆排序是一种选择排序（**正确**）。

32.对长度为8的表，作2路归并排序，关键字之间最多需要21次比较（**错误**）。

33.快速排序方法的每一趟都能将一个元素把它放到最终的位置上（**正确**）。

34.因为堆排序的算法时间复杂度为O(n㏒n)，冒泡排序的算法复杂度为O(n2 )，所以堆排序一定比冒泡排序的速度快（**错误**）。

35.对有n个记录的表作直接插入排序，在最坏的情况下，需比较关键字（不含与哨兵的比较）的次数为n(n-1)/2（**正确**）。

36.在快速排序、堆排序和归并排序中，快速排序需要的辅助空间最多（**错误**）。

37.如果冒泡排序的某趟过程中没有出现数据交换情况，那么说明关键字序列已经有序（**正确**）。

38.排序的稳定性是指排序算法中的比较次数保持不变，且算法能够终止（**错误**）。

39.在初始数据表已经有序时，快速排序算法的时间复杂度为O(n㏒n )（**错误**）。

40.如果关键字序列是堆，则关键字序列对应的二叉树是一棵二叉排序树（**错误**）。

41.在用堆排序算法排序时，如果要进行增序排序，则需要采用“大根堆”（**正确**）。

42.‏在任何情况下，归并排序都比简单插入排序快（**错误**）。

43.‎排序要求数据一定要以顺序方式存储（**错误**）。

44.直接选择排序的比较次数与关键字序列的初始状态无关（**正确**）。

45.因为接插入排序是稳定的，而Shell 排序是调用若干趟直接插入排序，所以也是稳定的（**错误**）。

46.‎以中序方式遍历一个堆序列对应的二叉树，则得到一个有序序列（**错误**）。

47.‍二路归并排序的核心操作是把两个有序序列合并为一个有序序列（**正确**）。

48.如果关键字序列采用单链表存储，那么基数排序过程可以避免大量数据移动（**正确**）。

49.基数排序是一种基于最高位优先(MSD)的多关键字排序法（**错误**）。

50.基于“比较”运算的排序算法，其时间复杂度的下界为O(n㏒n)（**正确**）。

以关键码序列（503，087，512，061，908，170，897，275，653，426）为例，手工执行以下(升序)排序算法，写出每个算法前两趟排序结束时的关键码状态（每2各关键码间用1个空格分隔，第1个前面，最后1个后面无空格）。 （1）直接插入排序的第一趟结束时关键码状态：▁▁▁； （2）直接插入排序的第二趟结束时关键码状态：▁▁▁； （3）希尔排序的第一趟结束时关键码状态：▁▁▁；(增量d[1]=5) （4）希尔排序的第二趟结束时关键码状态：▁▁▁；(增量d[2]=2)

以关键码序列（503，087，512，061，908，170，897，275，653，426）为例，手工执行以下(升序)排序算法，写出每个算法前两趟排序结束时的关键码状态（每2各关键码间用1个空格分隔，第1个前面，最后1个后面无空格）。 （1）快速排序的第一趟结束时关键码状态：▁▁▁； （2）快速排序的第二趟结束时关键码状态：▁▁▁； （3）序列初始化成大顶堆得到的关键码状态：▁▁▁； （4）堆排序的第一趟结束时关键码状态：▁▁▁； （5）堆排序的第二趟结束时关键码状态：▁▁▁。

以关键码序列（503，087，512，061，908，170，897，275，653，426）为例，手工执行以下(升序)排序算法，写出每个算法前两趟排序结束时的关键码状态（每2各关键码间用1个空格分隔，第1个前面，最后1个后面无空格）。

（1）归并排序的第一趟结束时关键码状态：▁▁▁；

（2）归并插入排序的第二趟结束时关键码状态：▁▁▁；

（3）基数排序的第一趟结束时关键码状态：▁▁▁；

（4）基数排序的第二趟结束时关键码状态：▁▁▁。