6--排序

【本节目标】

- 1.排序的概念及意义
- 2.直接插入和希尔排序的实现及分析
- 3.直接选择和堆排序的实现及分析
- 4.冒泡和快速排序的实现及分析
- 5.归并排序的实现及分析
- 6.排序的性能稳定性等性质的优劣对比
- 7.LeetCode OJ题 排序数组

1.排序的概念及其运用

1.1排序的概念

排序：所谓排序，就是使一串记录，按照其中的某个或某些关键字的大小，递增或递减的排列起来的操作。

稳定性：假定在待排序的记录序列中，存在多个具有相同的关键字的记录，若经过排序，这些记录的相对次序保持不变，即在原序列中，r[i]=r[j]，且r[i]在r[j]之前，而在排序后的序列中，r[i]仍在r[j]之前，则称这种排序算法是稳定的；否则称为不稳定的。

内部排序：数据元素全部放在内存中的排序。

外部排序：数据元素太多不能同时放在内存中，根据排序过程的要求不能在内外存之间移动数据的排序。

1.2排序运用

排名	院校名称	总分	类型	所在地	批次	历年排名
1	北京大学	100	综合类	北京	本科一批	
2	清华大学	99.58	理工类	北京	本科一批	
3	浙江大学	82.56	综合类	浙江	本科一批	
4	复旦大学	82.17	综合类	上海	本科一批	
5	中国人民大学	81.51	综合类	北京	本科一批	
6	上海交通大学	81.5	综合类	上海	本科一批	
7	武汉大学	81.49	综合类	湖北	本科一批	
8	南京大学	80.7	综合类	江苏	本科一批	
9	解放军国防科学技术大学	80.31	军事类	湖南	本科一批	
10	中山大学	76.16	综合类	广东	本科一批	
11	吉林大学	75.99	综合类	吉林	本科一批	
12	华中科技大学	75.04	综合类	湖北	本科一批	
13	四川大学	74.5	综合类	四川	本科一批	
14	天津大学	73.54	理工类	天津	本科一批	
15	南开大学	73.5	综合类	天津	本科一批	
16	西安交通大学	73.5	综合类	陕西	本科一批	

1.3 常见的排序算法


```
// 排序实现的接口

// 插入排序
void InsertSort(int* a, int n);

// 希尔排序
void ShellSort(int* a, int n);

// 选择排序
```

```
void SelectSort(int* a, int n);

// 堆排序
void AdjustDwon(int* a, int n, int root);
void HeapSort(int* a, int n);

// 冒泡排序
void BubbleSort(int* a, int n)

// 快速排序
void QuickSort(int* a, int left, int right);

// 归并排序
void MergeSort(int* a, int n)

// 测试排序的性能对比
void TestOP()
{
    srand(time(0));
    const int N = 100000;
    int* a1 = (int*)malloc(sizeof(int)*N);
    int* a2 = (int*)malloc(sizeof(int)*N);
    int* a3 = (int*)malloc(sizeof(int)*N);
    int* a4 = (int*)malloc(sizeof(int)*N);
    int* a5 = (int*)malloc(sizeof(int)*N);
    int* a6 = (int*)malloc(sizeof(int)*N);

    for (int i = 0; i < N; ++i)
    {
        a1[i] = rand();
        a2[i] = a1[i];
        a3[i] = a1[i];
        a4[i] = a1[i];
        a5[i] = a1[i];
        a6[i] = a1[i];

    }

    int begin1 = clock();
    InsertSort(a1, N);
    int end1 = clock();

    int begin2 = clock();
    ShellSort(a2, N);
    int end2 = clock();

    int begin3 = clock();
    SelectSort(a3, N);
    int end3 = clock();

    int begin4 = clock();
    HeapSort(a4, N);
    int end4 = clock();

    int begin5 = clock();
    QuickSort(a5, 0, N-1);
    int end5 = clock();
```

```
    int begin6 = clock();
    MergeSort(a6, N);
    int end6 = clock();

    printf("InsertSort:%d\n", end1 - begin1);
    printf("ShellSort:%d\n", end2 - begin2);
    printf("SelectSort:%d\n", end3 - begin3);
    printf("HeapSort:%d\n", end4 - begin4);
    printf("QuickSort:%d\n", end5 - begin5);
    printf("MergeSort:%d\n", end6 - begin6);

    free(a1);
    free(a2);
    free(a3);
    free(a4);
    free(a5);
    free(a6);
}
```

1. 插入排序

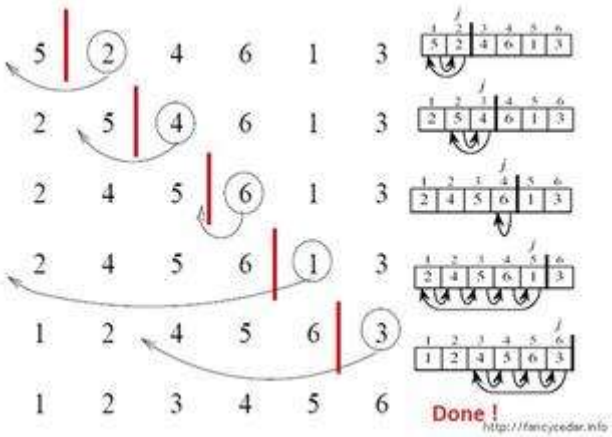
1.1基本思想:

直接插入排序是一种简单的插入排序法，其基本思想是：把待排序的记录按其关键码值的大小逐个插入到一个已经排好序的有序序列中，直到所有的记录插入完为止，得到一个新的有序序列 。

实际中我们玩扑克牌时，就用了插入排序的思想

1.2直接插入排序:

当插入第i(i>=1)个元素时，前面的array[0],array[1],...,array[i-1]已经排好序，此时用array[i]的排序码与array[i-1],array[i-2],...的排序码顺序进行比较，找到插入位置即将array[i]插入，原来位置上的元素顺序后移

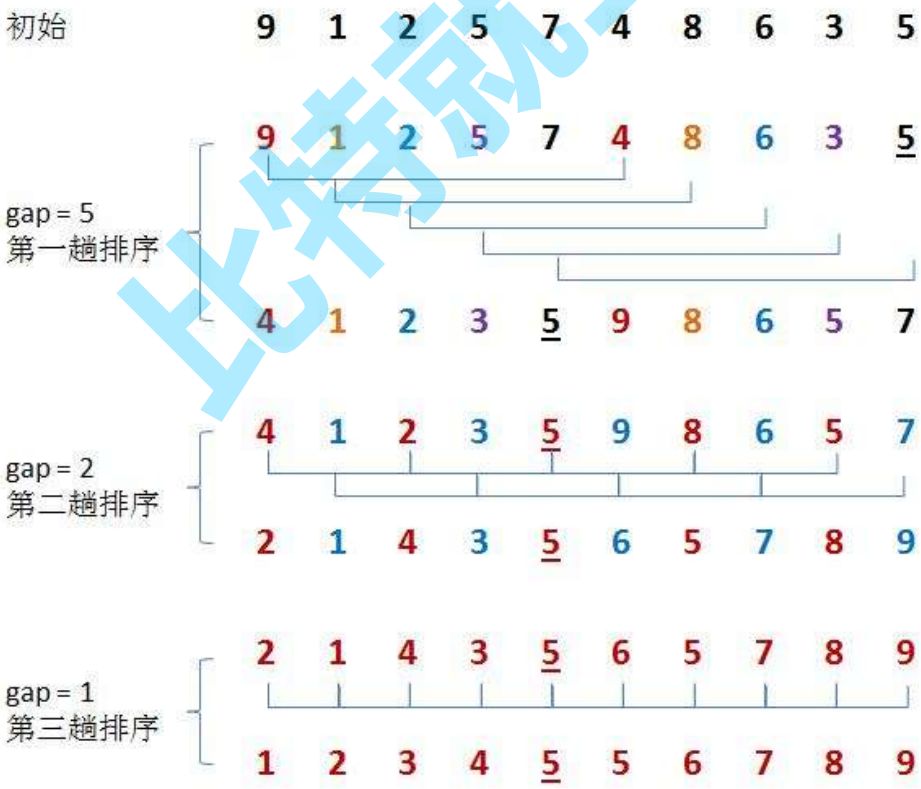


直接插入排序的特性总结：

1. 元素集合越接近有序，直接插入排序算法的时间效率越高
2. 时间复杂度：O(N^2)
3. 空间复杂度：O(1)，它是一种稳定的排序算法
4. 稳定性：稳定

1.3 希尔排序(缩小增量排序)

希尔排序法又称缩小增量法。希尔排序法的基本思想是：先选定一个整数，把待排序文件中所有记录分成个组，所有距离为的记录分在同一组内，并对每一组内的记录进行排序。然后，取，重复上述分组和排序的工作。当到达=1时，所有记录在统一组内排好序。



希尔排序的特性总结：

1. 希尔排序是对直接插入排序的优化。
2. 当gap > 1时都是预排序，目的是让数组更接近于有序。当gap == 1时，数组已经接近有序的了，这样就会很快。这样整体而言，可以达到优化的效果。我们实现后可以进行性能测试的对比。

3. 希尔排序的时间复杂度不好计算，需要进行推导，推导出来平均时间复杂度： O(N^1.3—N^2)
4. 稳定性：不稳定

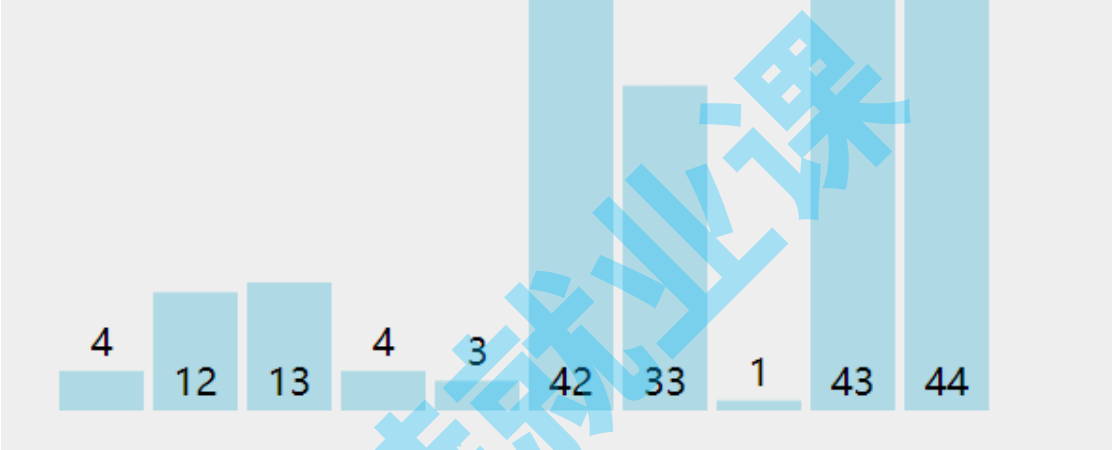
2选择排序

2.1基本思想:

每一次从待排序的数据元素中选出最小（或最大）的一个元素，存放在序列的起始位置，直到全部待排序的数据元素排完 。

2.2 直接选择排序:

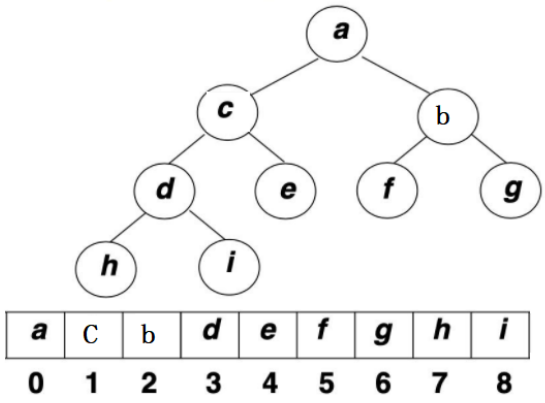
- 在元素集合array[i]--array[n-1]中选择关键码最大(小)的数据元素
- 若它不是这组元素中的最后一个(第一个)元素，则将它与这组元素中的最后一个（第一个）元素交换
- 在剩余的array[i]--array[n-2]（array[i+1]--array[n-1]）集合中，重复上述步骤，直到集合剩余1个元素



直接选择排序的特性总结:

1. 直接选择排序思考非常好理解，但是效率不是很好。实际中很少使用
2. 时间复杂度：O(N^2)
3. 空间复杂度：O(1)
4. 稳定性：不稳定

2.3 堆排序



堆的逻辑结构是一颗完全二叉树

堆的物理结构是一个数组

通过下标父子节点关系

```
parent = leftchild*2+1
parent = leftchild*2+2

child = (parent-1)/2
```

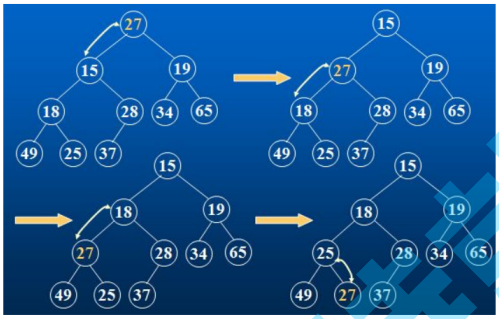
➢ 堆的两个特性

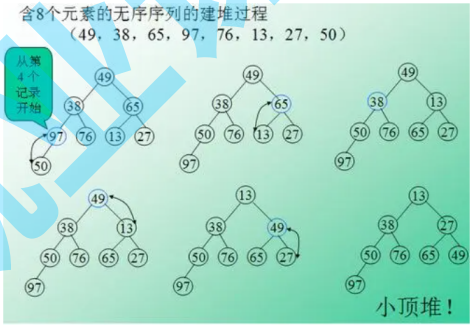
☞ 结构性：用数组表示的完全二叉树；

☞ 有序性：任一结点的关键字是其子树所有结点的最大值(或最小值)

- ❑ “最大堆(MaxHeap)”,也称“大顶堆”：最大值
- ❑ “最小堆(MinHeap)”,也称“小顶堆”：最小值

建堆的过程





2. 自底向上的建堆方式

该建堆方式是从倒数第二层的节点（叶子节点的上一层）开始，从右向左，从下到上的向下进行调整。

同样的，假设该堆为满二叉树，堆高为 h 。同样的，假设每层高度为 h_i ，每层结点数为 n_i ，则建堆复杂度为 $t(n)=\sum_{i=1}^{h-1} n_i \cdot h_i$ ，则有

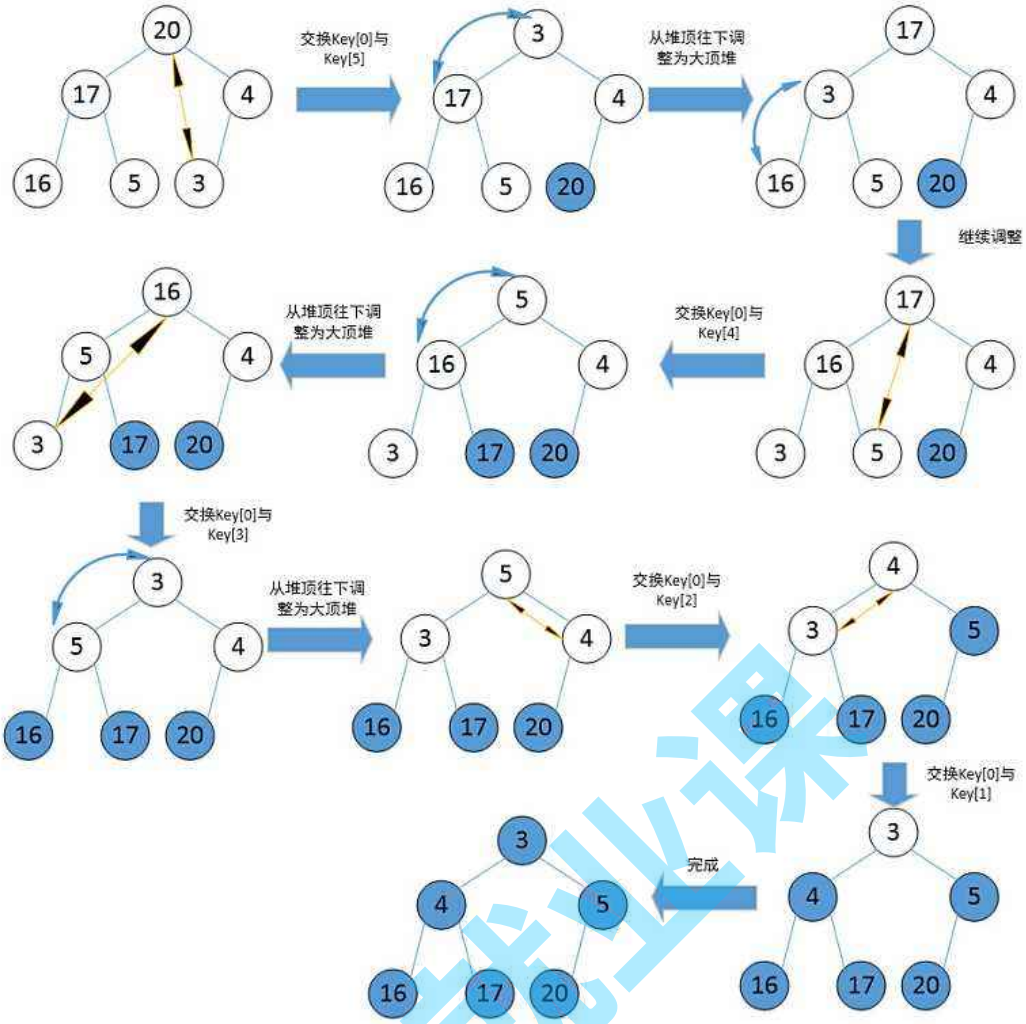
$$t(n)=1\times(h-1)+2\times(h-2)+4\times(h-3)+\ldots+2^{h-2}\times 1$$
$$=2^0\times(h-1)+2^1\times(h-2)+2^2\times(h-3)+\ldots+2^{h-2}\times 1$$

同样的，该数列和为差比数列。因此，可以用错位相减法，得到时间复杂度为

$$t(n)=2^h-h-1=n-log(n+1).$$

即，时间复杂度为 $T(n)=n$.

堆排序(Heapsort)是指利用堆积树（堆）这种数据结构所设计的一种排序算法，它是选择排序的一种。它是通过堆来进行选择数据。**需要注意的是排升序要建大堆，排降序建小堆。**



堆排序的特性总结：

1. 堆排序使用堆来选数，效率就高了很多。
2. 时间复杂度：O(N*logN)
3. 空间复杂度：O(1)
4. 稳定性：不稳定

3. 交换排序

基本思想：所谓交换，就是根据序列中两个记录键值的比较结果来对换这两个记录在序列中的位置，交换排序的特点是：将键值较大的记录向序列的尾部移动，键值较小的记录向序列的前部移动。

3.1冒泡排序

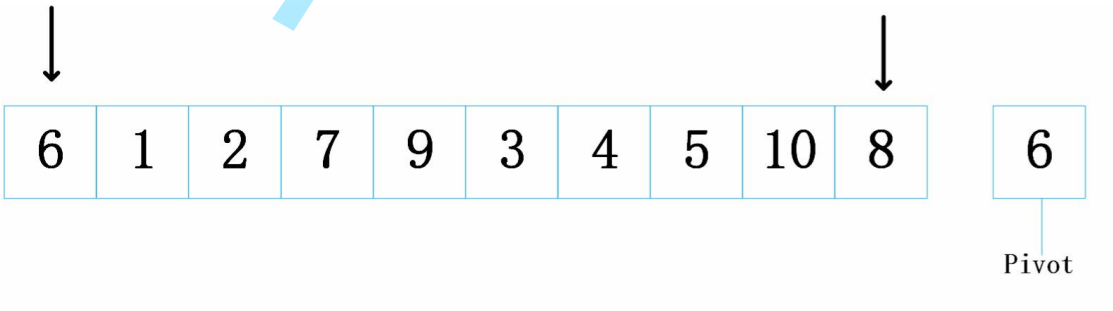
冒泡排序的特性总结：

1. 冒泡排序是一种非常容易理解的排序
2. 时间复杂度：O(N^2)
3. 空间复杂度：O(1)
4. 稳定性：稳定

3.2 快速排序

快速排序是Hoare于1962年提出的一种二叉树结构的交换排序方法，其基本思想为：任取待排序元素序列中的某元素作为基准值，按照该排序码将待排序集合分割成两子序列，左子序列中所有元素均小于基准值，右子序列中所有元素均大于基准值，然后最左右子序列重复该过程，直到所有元素都排列在相应位置上为止。

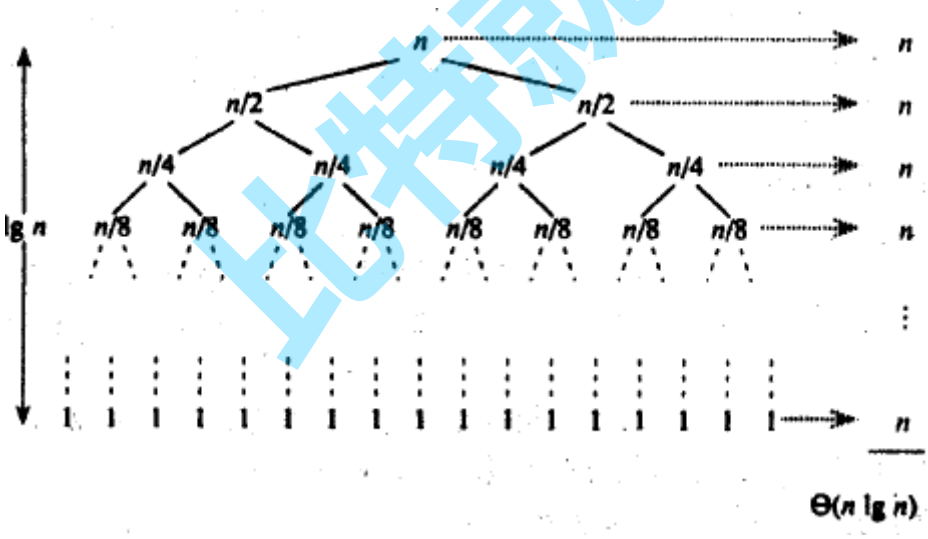
将区间按照基准值划分为左右两半部分的常见方式有：



整体实现思想：

快速排序的特性总结：

1. 快速排序整体的综合性能和使用场景都是比较好的，所以才敢叫**快速**排序
2. 时间复杂度：O(N*logN)

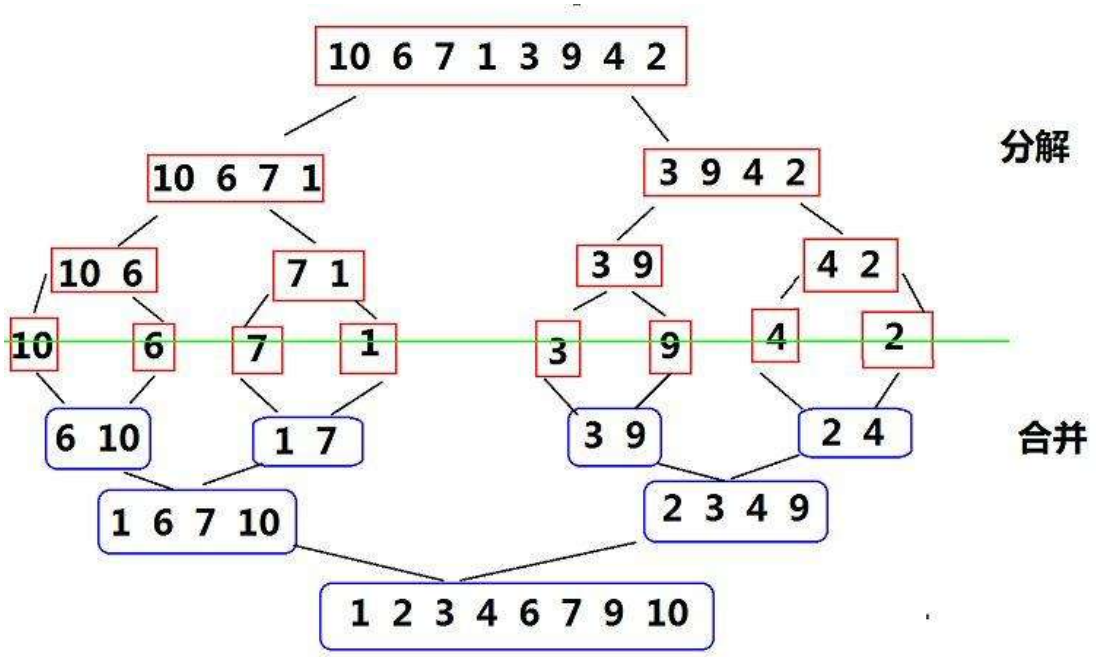


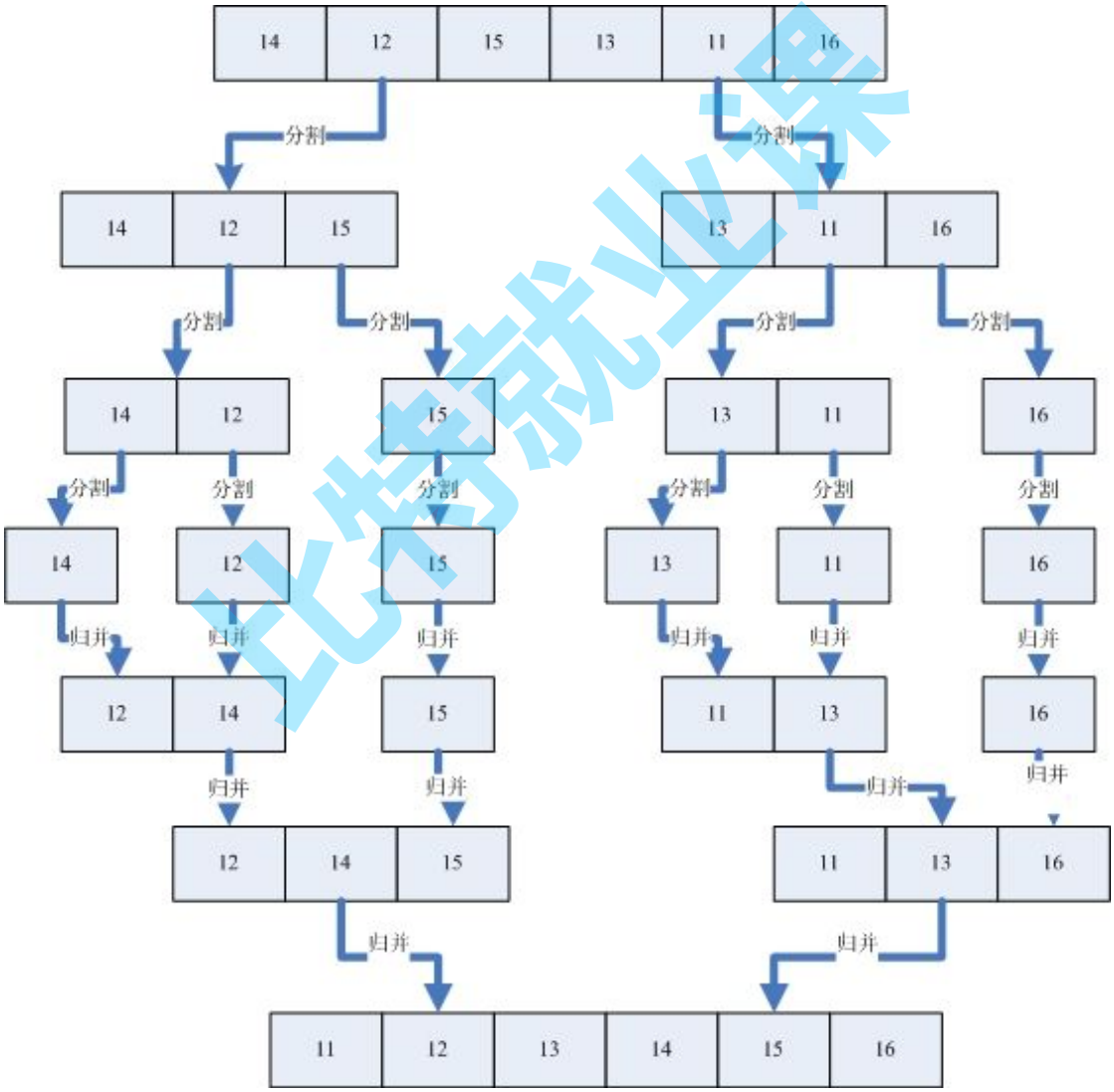
3. 空间复杂度：O(logN)
4. 稳定性：不稳定

4.归并排序

基本思想：

归并排序（MERGE-SORT）是建立在归并操作上的一种有效的排序算法,该算法是采用分治法（Divide and Conquer）的一个非常典型的应用。将已有序的子序列合并，得到完全有序的序列；即先使每个子序列有序，再使子序列段间有序。若将两个有序表合并成一个有序表，称为二路归并。 归并排序核心步骤：





归并排序的特性总结：

1. 归并的缺点在于需要O(N)的空间复杂度，归并排序的思考更多的是解决在磁盘中的外排序问题。
2. 时间复杂度：O(N*logN)
3. 空间复杂度：O(N)
4. 稳定性：稳定

5.排序算法复杂度及稳定性分析

排序方法	平均情况	最好情况	最坏情况	辅助空间	稳定性
冒泡排序	$O(n^2)$	$O(n)$	$O(n^2)$	O(1)	稳定
选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	O(1)	不稳定
插入排序	$O(n^2)$	$O(n)$	$O(n^2)$	O(1)	稳定
希尔排序	$O(n \log n)$~ $O(n^2)$	$O(n^{1.3})$	$O(n^2)$	O(1)	不稳定
堆排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	O(1)	不稳定
归并排序	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$	$O(n)$	稳定
快速排序	$O(n \log n)$	$O(n \log n)$	$O(n^2)$	$O(\log n)$~ $O(n)$	不稳定

6.选择题练习

1. 快速排序算法是基于（　　）的一个排序算法。
A分治法
B贪心法
C递归法
D动态规划法

2.对记录（54,38,96,23,15,72,60,45,83）进行从小到大的直接插入排序时，当把第8个记录45插入到有序表时，为找到插入位置需比较（ ）次？（采用从后往前比较）
A 3
B 4
C 5
D 6

3.以下排序方式中占用O（n）辅助存储空间的是
A 简单排序
B 快速排序
C 堆排序
D 归并排序

4.下列排序算法中稳定且时间复杂度为O(n2)的是（ ）
A 快速排序
B 冒泡排序
C 直接选择排序
D 归并排序

5.关于排序，下面说法不正确的是
A 快排时间复杂度为O(N*logN)，空间复杂度为O(logN)
B 归并排序是一种稳定的排序,堆排序和快排均不稳定
C 序列基本有序时，快排退化成冒泡排序，直接插入排序最快
D 归并排序空间复杂度为O(N)， 堆排序空间复杂度的为O(logN)

6.下列排序法中，最坏情况下时间复杂度最小的是（ ）
A 堆排序
B 快速排序
C 希尔排序
D 冒泡排序

7.设一组初始记录关键字序列为(65,56,72,99,86,25,34,66)，则以第一个关键字65为基准而得到的一趟快速排序结果是（）
A 34，56，25，65，86，99，72，66
B 25，34，56，65，99，86，72，66
C 34，56，25，65，66，99，86，72

D 34，56，25，65，99，86，72，66

答案：
1.A
2.C
3.D
4.B
5.D
6.A
7.A

7.LeetCode OJ题

排序数组：https://leetcode-cn.com/problems/sort-an-array/ 试试哪个写排序可以跑过这个oj测试