数据结构与算法（八）

张铭 主讲

采用教材：张铭，王腾蛟，赵海燕 编写
高等教育出版社，2008. 6（“十一五”国家级规划教材）

http://www.jpk.pku.edu.cn/pkujpk/course/sjjg

大 纲

- 8.1 排序问题的基本概念
- 8.2 插入排序 (Shell 排序)
- 8.3 选择排序 (堆排序)
- 8.4 交换排序
  - 8.4.1 冒泡排序
  - 8.4.2 快速排序
- 8.5 归并排序
- **8.6 分配排序**和索引排序
- 8.7 排序算法的时间代价
- 内排序知识点总结

8.6 分配排序和基数排序

- 不需要进行纪录之间两两比较
- 需要事先知道记录序列的一些具体情况

8.6.1 桶式排序

- 事先知道序列中的记录都位于某个小区间段 [0, m) 内
- 将具有相同值的记录都分配到同一个桶中，然后依次按照编号从桶中取出记录，组成一个有序序列

8.6.1 桶式排序

待排数组： 7 3 8 9 6 1 8′ 1′ 2

每个桶 count：

0	1	2	3	4	5	6	7	8	9
0	2	1	1	0	0	1	1	2	1

＋ ＋ ＋ ＋

前若干桶的累计count：

0	1	2	3	4	5	6	7	8	9
0	2	3	4	4	4	5	6	8	9

8.6.1 桶式排序

桶排序示意

待排数组： 7 3 8 9 6 1 8′ 1′ 2

每个桶count：

0	1	2	3	4	5	6	7	8	9
0	2	1	1	0	0	1	1	2	1

前若干桶的累计count：

0	0	2	3	4	4	4	5	6	8

收集：

0	1	2	3	4	5	6	7	8
1	1’	2	3	6	7	8	8’	9

桶式排序算法

```
template <class Record> void BucketSort(Record Array[], int n, int max) {
    Record *TempArray = new Record[n];   // 临时数组
    int *count = new int[max];           // 桶容量计数器
    int i;
    for (i = 0; i < n; i++)              // 把序列复制到临时数组
        TempArray[i] = Array[i];
    for (i = 0; i < max; i++)            // 所有计数器初始都为0
        count[i] = 0;
    for (i = 0; i < n; i++)              // 统计每个取值出现的次数
        count[Array[i]]++;
    for (i = 1; i < max; i++)            // 统计小于等于i的元素个数
        count[i] = count[i-1]+count [i];  // c [i]记录i+1的起址
    for (i = n-1; i >= 0; i--)           // 尾部开始，保证稳定性
        Array[--count[TempArray[i]]] = TempArray[i];
}
```

8.6.1 桶式排序

算法分析

- 数组长度为 n, 所有记录区间 $[0, m)$ 上
- 时间代价：
  - 统计计数：$\Theta(n+m)$ ， 输出有序序列时循环 n 次
  - 总的时间代价为 $\Theta(m+n)$
  - 适用于 m 相对于 n 很小的情况
- 空间代价：
  - m 个计数器，长度为 n 的临时数组，$\Theta(m+n)$
- 稳定

思 考

1. 桶排事先知道序列中的记录都位于某个小区间段 [0，m) 内。m 多大合适？超过这个范围怎么办？
2. 桶排中，count 数组的作用是什么？为什么桶排要从后往前收集?