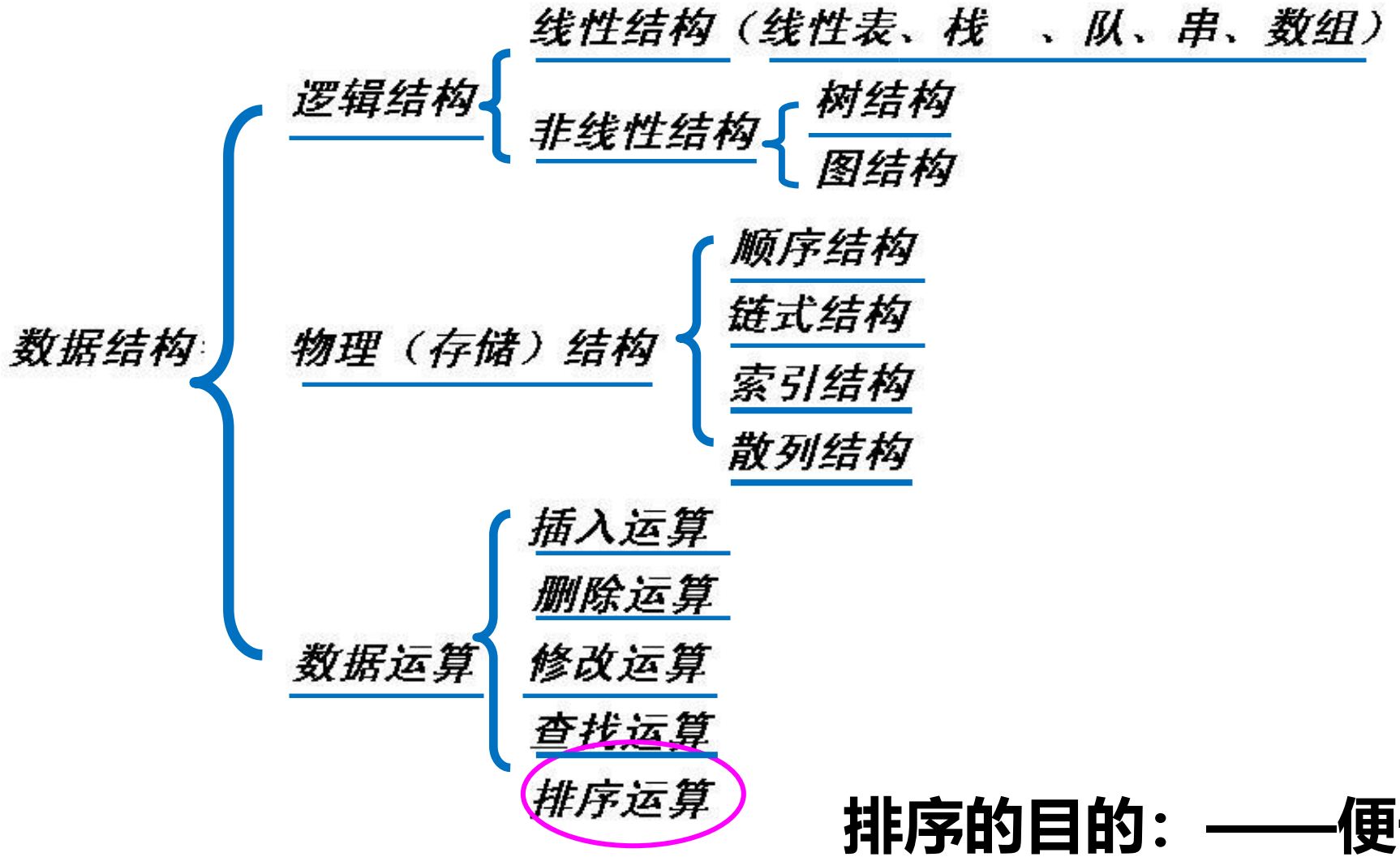
数据结构课程的内容



排序的目的：——便于查找！

插入	• 直接插入排序； 二分插入排序 • 希尔排序
选择	• 直接选择排序 • 堆排序
交换	• 冒泡排序 • 快速排序
分配	• 基数排序
归并	• 两路归并 • 多路归并

排序是将“无序”的记录序列调整为“有序”的记录序列

8.1 排序的基本概念

设有记录序列：{ R0、R2 ... Rn-1 } 其相应的关键字序列为： { K0、K2 ... Kn-1 };

已知 $K_i=K_j$ $j>i$

稳定排序：

不稳定排序：

排序后

R_i 领先于 R_j

R_j 领先于 R_i

例如：按递增顺序排列

49(1) 38 65 97 76 13 27 49(2) → 13 27 38 49(1) 49(2) 65 76 97

排序后具有相同关键字的记录的相对次序保持不变

8.1 排序的基本概念

排序中的基本操作：关键码大小的比较和记录的移动

评价排序算法好坏的标准

- 第一是执行算法所需的时间
  - 比较次数
  - 移动次数
- 执行算法所需要的附加空间；
- 算法本身的复杂程度也是考虑的一个因素

按排序算法的时间复杂度不同，可分为3类：

- 简单的排序算法：时间效率低$O(n^2)$
- 先进的排序算法：时间效率高$O(n\log_2 n)$
- 基数排序算算法：时间效率高$O(d\times n)$

8.1 排序的基本概念

内部排序和外部排序

- 内（部）排序：整个的排序过程不需要访问外存便能完成
- 外（部）排序：排序的记录数量很大，整个序列的排序过程需要内外存的交换

待排序的记录的存储结构

- 顺序存储
- 链式存储
- 索引存储

8.1 排序的基本概念

本章待排序的记录采用顺序存储结构，类型定义如下：

```
1  typedef int KeyType;
2  typedef int InfoType;
3  typedef struct RECORDTYPE_STRU
4  {
5          KeyType key;       //关键字
6          InfoType otherInfo; //其余数据信息
7  }RecordType;
8  typedef struct SORTARRAY_STRU
9  {
10         int cnt;    // 记录排序数组中的元素个数
11         RecordType *recordArr; //指向一维数组的指针
12 }SortArr;
```

8.1 排序的基本概念

排序过程的一个主要操作是交换记录，为了方便后面算法的描述，将该操作单独写成一个函数，如算法8-1所示。

```
1  //交换两个记录
2  void  Swap(SortArr* sortArr, int i, int j)
3  {
4          KeyType temp;
5          temp = sortArr->recordArr[i].key;
6          sortArr->recordArr[i].key = sortArr->recordArr[j].key;
7          sortArr->recordArr[j].key = temp;
8  }
```

算法8-1