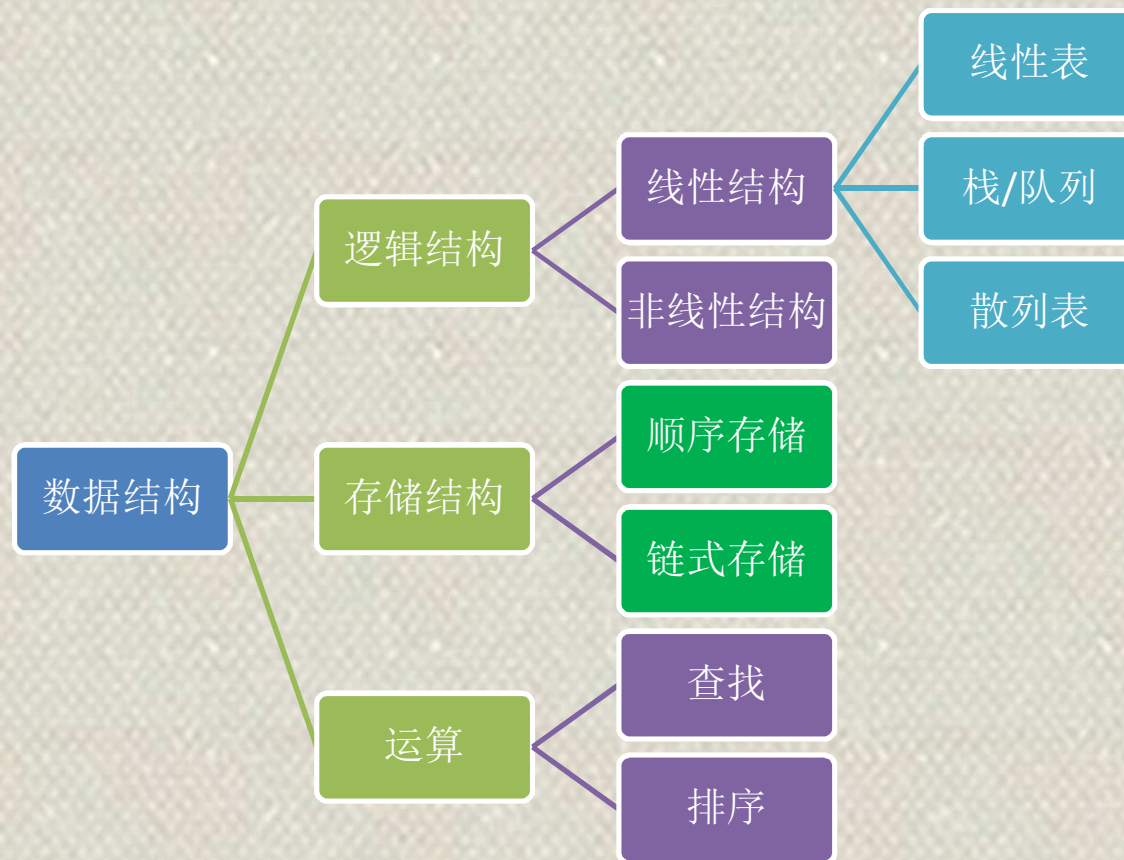


Ch.7 会应用到的数据结构

2017 / 11

数据结构



线性

- 数组
- 单（向）链表
- 双（向）链表
- 队列
- 栈
- 散列表

连续存储

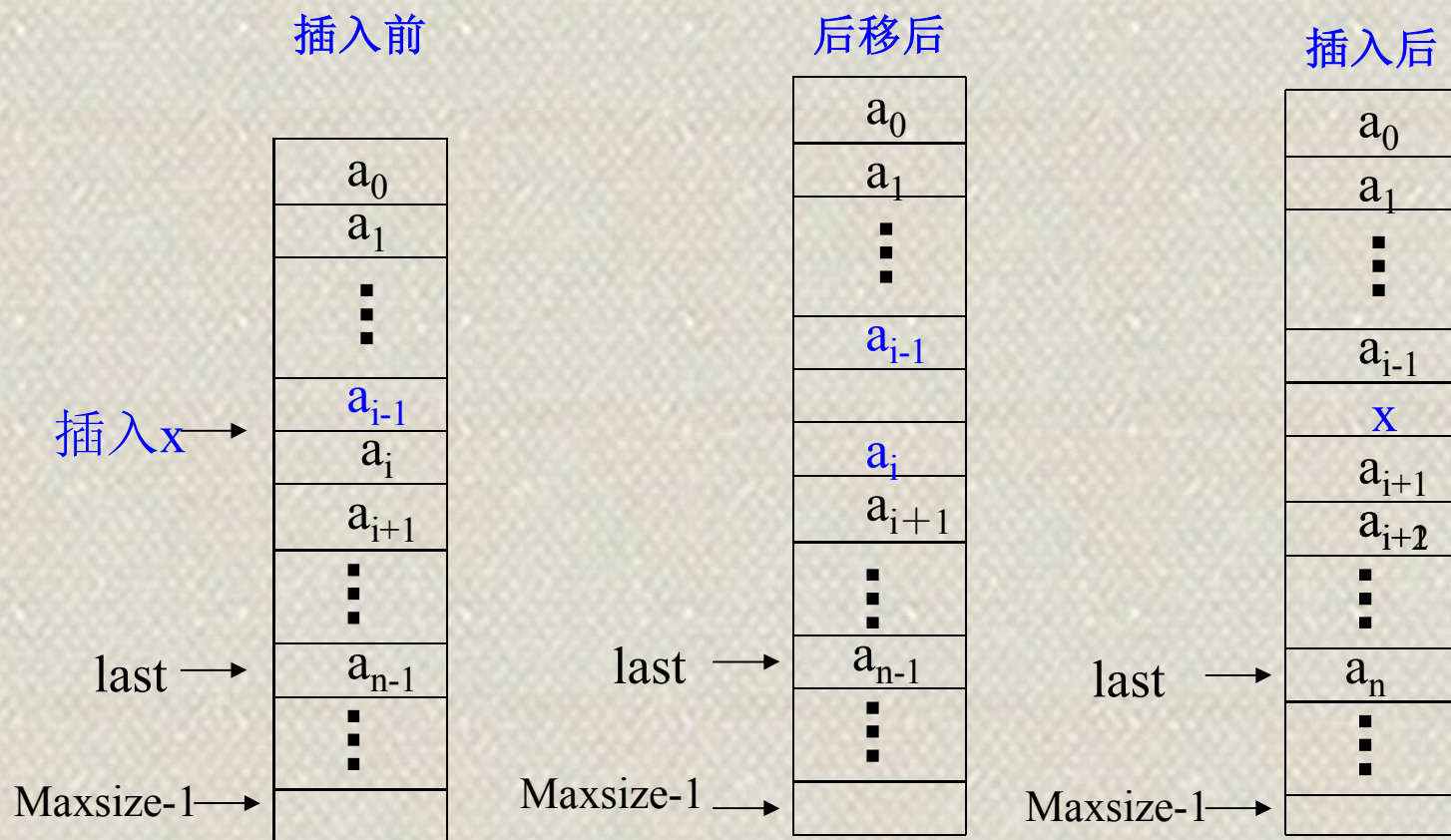
- 内容连续
- 随机访问很方便
- 插入或删除的维护麻烦
- 元素增加，扩充问题

存储地址	存储内容
L_0	元素1
L_0+1	元素2

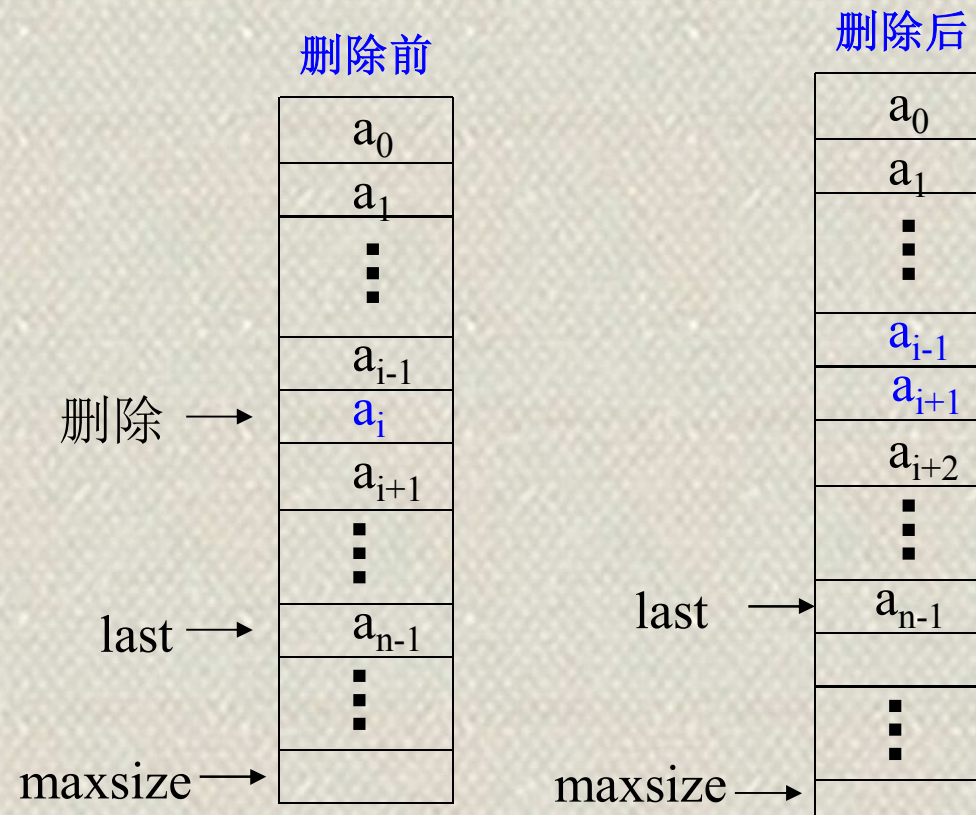
L_0+i-1	元素i

L_0+n-1	元素n

连续存储-插入

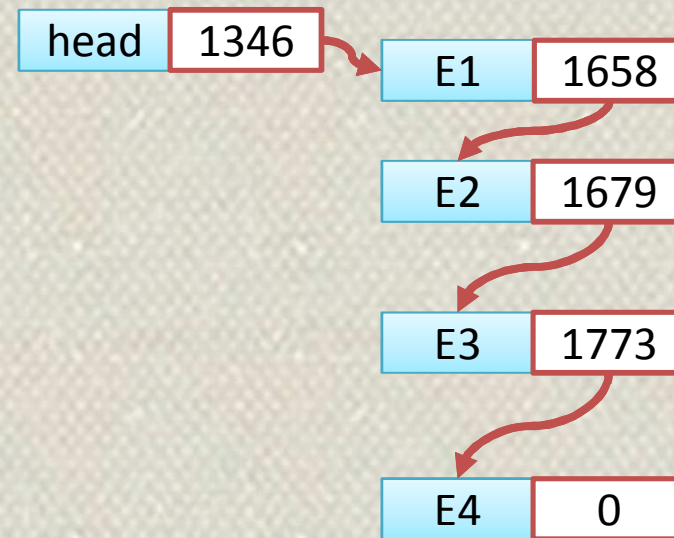


连续存储-删除



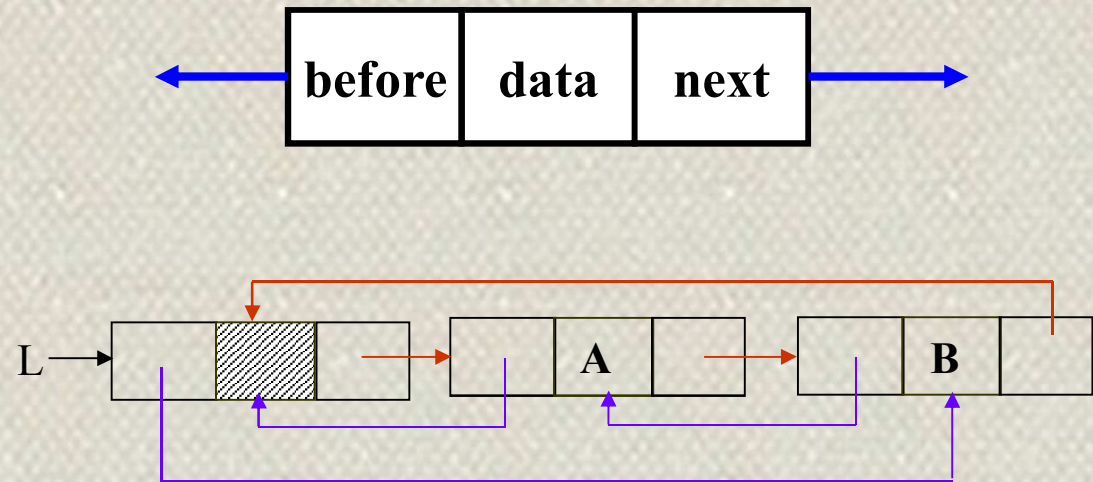
单链表

- 逻辑上相邻，物理上不相邻
- 不能随机访问（用循环计数才行）
- 方便扩展
- 方便操作（增/删）

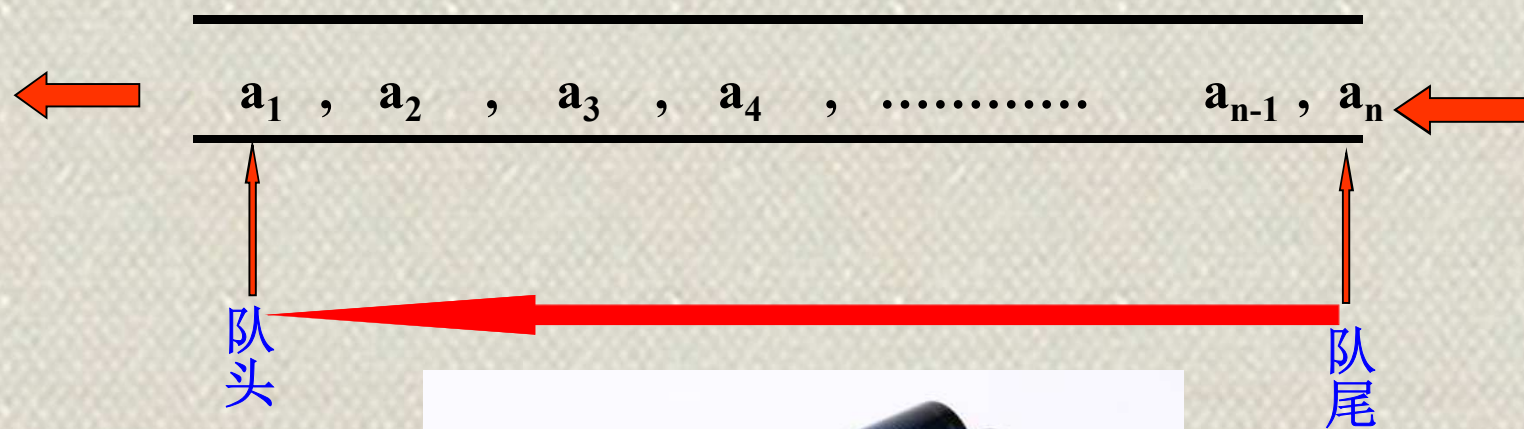


双链表

- 在每个结点中设置两个指针，一个指向后继，一个指向前驱。可直接确定一个结点的前驱和后继结点。可提高效率



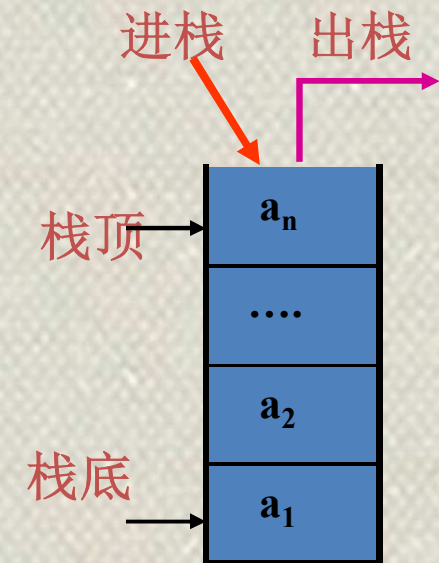
队列



栈

❖ **栈**：限定只能在表的一端进行插入和删除的特殊的线性表，此种结构称为**后进先出**。

- * 设栈 $s = (a_1, a_2, \dots, a_i, \dots, a_n)$
- * 其中 a_1 是栈底元素， a_n 是栈顶元素。
- * 栈顶 (top)：允许插入和删除的一端；
- * 约定 top 始终指向新数据元素将存放的位置。
- * 栈底 (bottom)：不允许插入和删除的一端。



散列表

- hash_map
- unordered_map
- $\text{Id} \% 13$ (一般是质数)

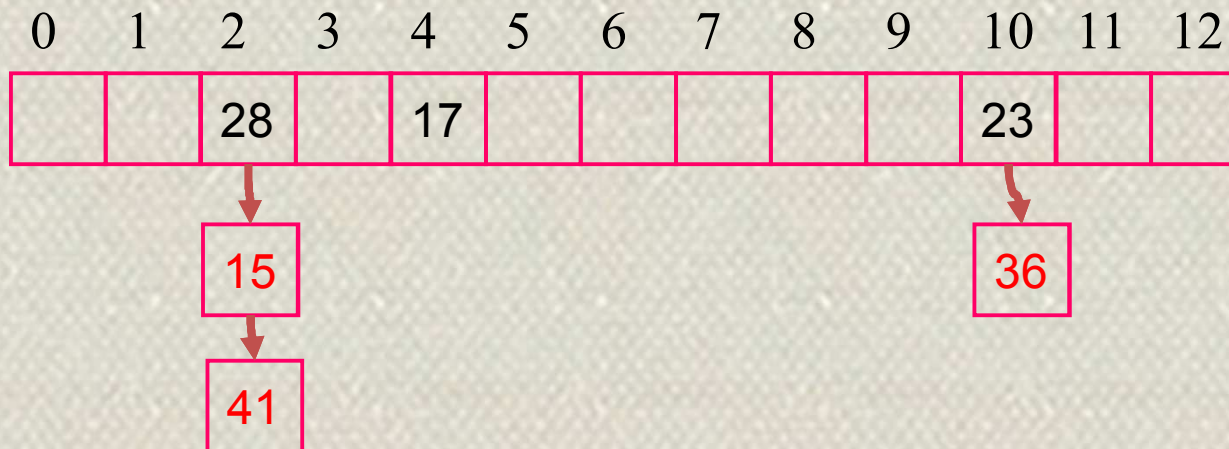
0	1	2	3	4	5	6	7	8	9	10	11	12
		28		17						23		

冲突的解决方法

- 线性探查（开放地址）
- 链表（桶）

Id % 13

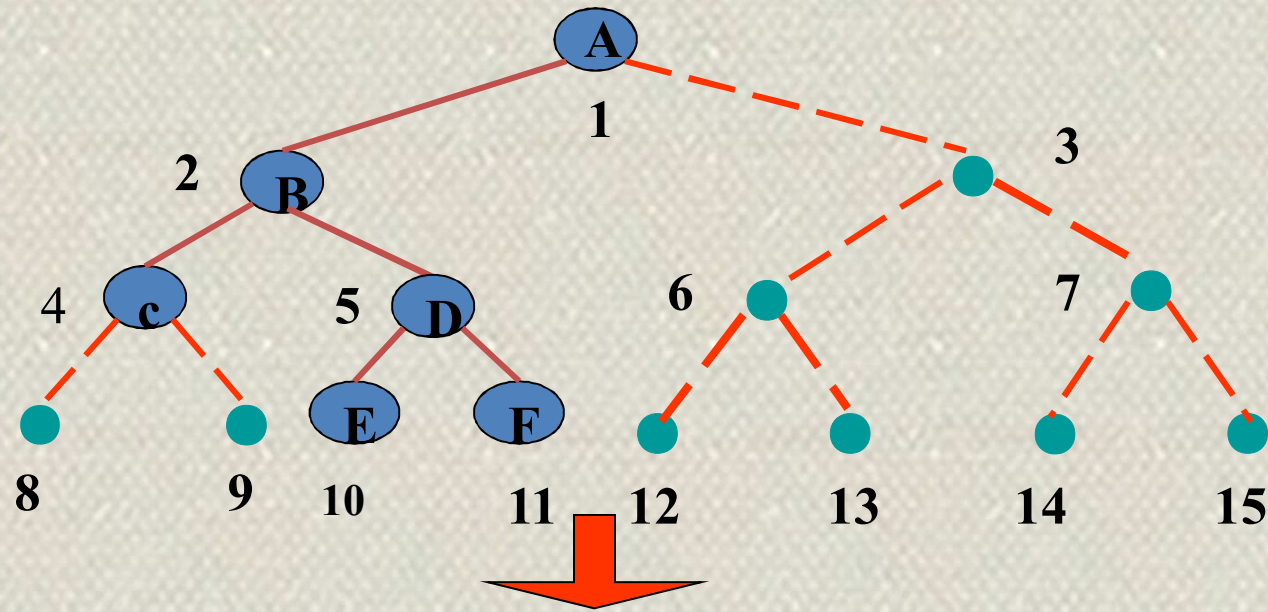
0	1	2	3	4	5	6	7	8	9	10	11	12
		28	15	17	41					23	36	



树和树的应用

- 二叉树
- 二叉排序树
- 哈夫曼树
- B树、B+树

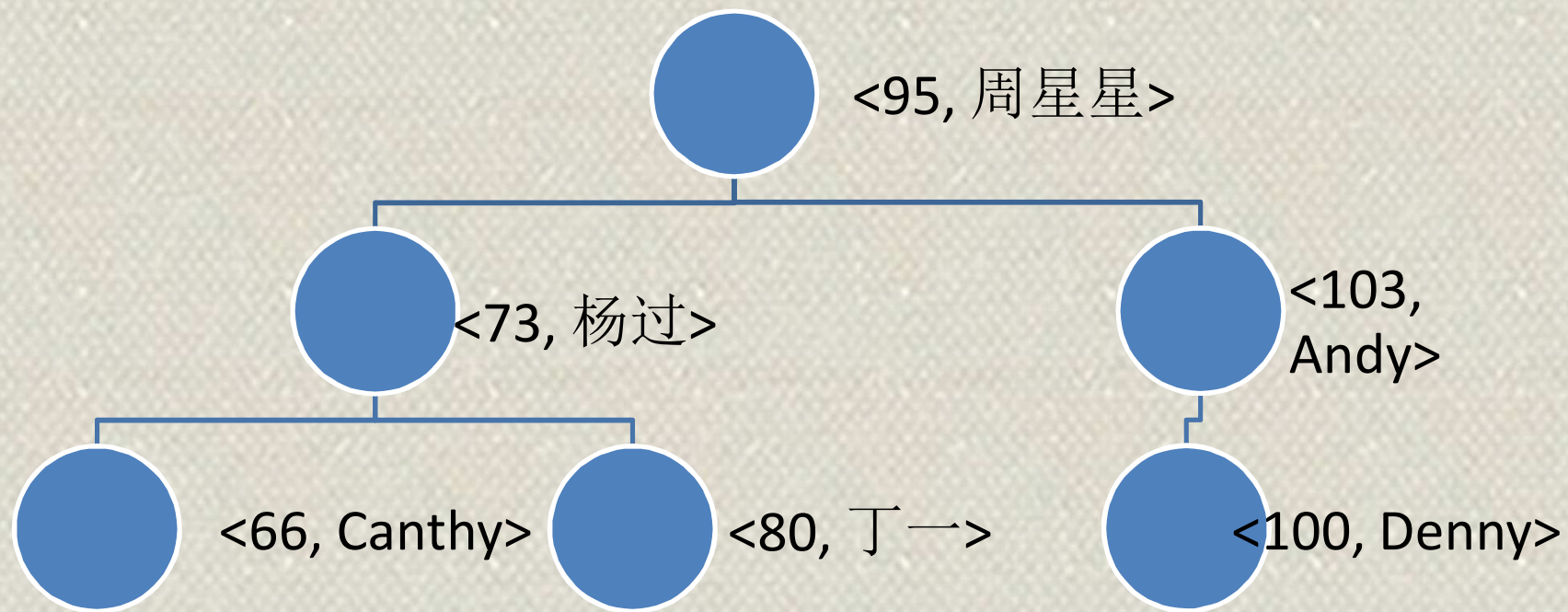
二叉树



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
T[16]		A	B	0	C	D	0	0	0	0	E	F	0	0	0	0

$$2^h - 1 = 2^4 - 1 = 15$$

二叉排序树



查找算法

- 线性查找
- 折半（二分）查找

排序算法

- 插入排序
- 快速排序
- 堆排序
- 归并排序