



第12周小结

1

线性表查找

① 线性表的存储结构

- 顺序表—静态查找表
- 链表—动态查找表

② 顺序表查找算法

● 顺序查找

● 折半查找

● 分块查找

} 成功情况下的ASL

不成功情况下的ASL



设有100个元素的有序表，采用折半查找方法，成功时最大的比较次数是（ ）。

A.25

B.50

C.10

D.7 ✓

成功时最大比较次数为 $\lceil \log_2(n+1) \rceil = \lceil \log_2 101 \rceil = 7$ 。



从19个元素中查找其中某个元素，如果最多进行5次元素之间的比较，则采用的查找方法只可能是（ ）。

A.折半查找 ✓

B.分块查找

C.顺序查找

D.都不可能

$n=19$ ，折半查找的元素最多比较次数 $=\lceil \log_2(n+1) \rceil = 5$ ，顺序查找和分块查找所需元素比较次数会更多。



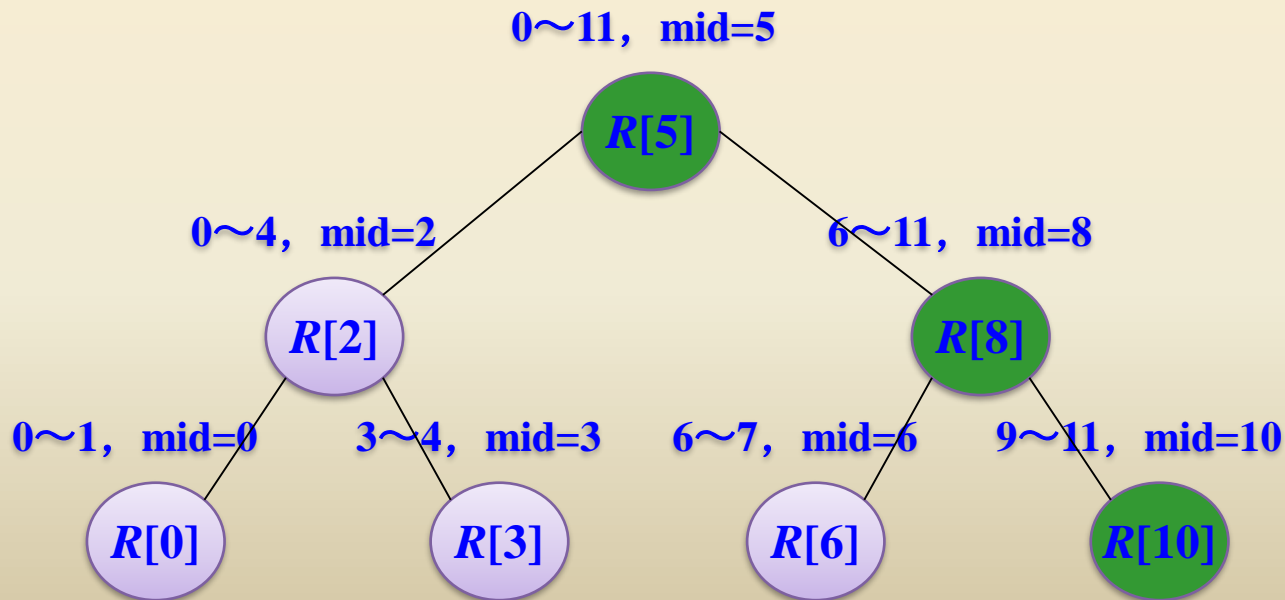
一个递增表为 $R[0..11]$ ，采用折半查找方法，在某次成功查找到指定的记录时，以下（）是可能的记录比较序列。

A. $R[0]$ 、 $R[5]$ 、 $R[2]$

B. $R[0]$ 、 $R[6]$ 、 $R[9]$

C. $R[5]$ 、 $R[8]$ 、 $R[10]$ ✓

D. $R[5]$ 、 $R[2]$ 、 $R[4]$



当采用分块查找时，数据的组织方式为（ ）。



- A. 数据分成若干块，每块内数据有序
- B. 数据分成若干块，每块内数据不必有序，但块间必须有序，每块内最大（或最小）的数据组成索引块 ✓
- C. 数据分成若干块，每块内数据有序，每块内最大（或最小）的数据组成索引块
- D. 数据分成若干块，每块中的数据个数必须相同

块内无序，块间有序！



设待查找元素为47，且已存入变量 k 中，如果在查找过程中，和 k 进行比较的元素依次是47、32、46、25、47，则所采用的查找方法（ ）。

A.是一种错误的方法

B.可能是分块查找 ✓

C.可能是顺序查找

D.可能是折半查找

- 顺序查找或折半查找 ⇨ 第一次比较成功时就会结束。
- 可能是分块查找，假设索引表是对块中最大元素进行索引，先和索引表中47比较找到相应块，然后到相应块（32、46、25、47）中查找。

2

树表查找

① 二叉排序树

二叉树结构 + BST特性



二叉排序树

基本运算：查找、插入、删除

二叉排序树重要属性



- 二叉排序树 \Rightarrow 中序序列是一个递增有序序列
- 二叉树 + 中序序列是一个递增有序序列 \Rightarrow 二叉排序树

- 二叉排序树中最小节点是中序序列的开始节点
- 二叉排序树中最小节点是根节点最左下节点

- 二叉排序树中最大节点是中序序列的尾节点
- 二叉排序树中最大节点是根节点最右下节点



用 n 个关键字构造的一棵二叉排序树，经过 i 次关键字比较成功找到的元素个数最多为（ ）。

A. i

B. 2^i

C. 2^{i-1} ✓

D. 2^i-1

二叉排序树中第 i 层最多有 2^{i-1} 个节点。



对于下列关键字序列，不可能构成某二叉排序树中一条查找路径的序列是（ ）。

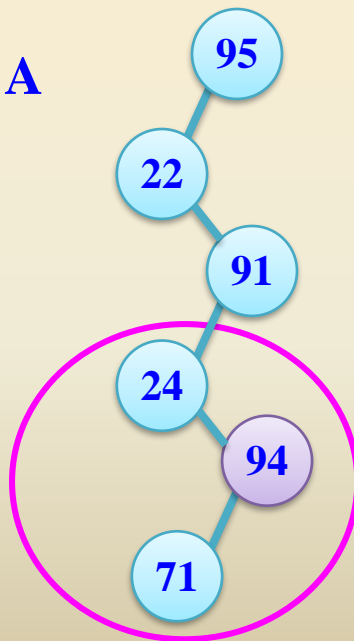
A. 95, 22, 91, 24, 94, 71 ✗

B. 92, 20, 91, 34, 88, 35

C. 21, 89, 77, 29, 36, 38

D. 12, 25, 71, 68, 33, 34

选项A





有一棵含有8个节点的二叉排序树，其节点值为A～H，以下（ ）是其后续遍历结果。

A.ADBCEGFH


B.BCAGEHFD

C.BCAEFDHG ✓

D.BDACEFHG

中序序列为ABCDEFGH，与每一个选项的后序序列构造二叉排序树，只有C可以构造出一棵二叉排序树。

② 平衡二叉树

二叉排序树 + 平衡特性  平衡二叉树

- 查找运算：

与二叉排序树 相同

- 插入运算：

按二叉排序树 方式插入 + 调整

- 删除运算：

按二叉排序树 方式删除 + 调整

调整方式：

- *LL*

- *RR*

- *LR*

- *RL*

平衡二叉树与二叉排序树的差别:

- 由 n 个关键字构造的二叉排序树高度为 $\lceil \log_2(n+1) \rceil \sim n$, 查找效率为 $O(\log_2 n) \sim O(n)$
- 由 n 个关键字构造的平衡二叉树高度为 $O(\log_2 n)$, 查找效率为 $O(\log_2 n)$



在含有12个节点的平衡二叉树上，查找关键字为35（存在该节点）的节点，则依次比较的关键字有可能是（ ）。

- A. 46, 36, 18, 20, 28, 35 B. 47, 37, 18, 27, 36
C. 27, 48, 39, 43, 37 D. 15, 45, 25, 35 ✓

$$N_1=1, N_2=2, N_h=N_{h-1}+N_{h-2}+1$$

$$N_3=4, N_4=7, N_5=12, \text{ 求出 } N_h=12 \text{ 时, } h=5。$$

也就是说，12个节点的平衡二叉树最大高度为5

选项A比较6次 ⇨ 错误；选项B、C比较5次而不成功 ⇨ 错误

③ B-树和B+树

m 阶B-树重要属性

- n 个关键字的节点有 $n+1$ 棵子树
- 内部节点关键字总数为 n ，外部节点个数为 $n+1$
- 内部节点最多关键字个数 $\text{Max} = m-1$
- 内部节点最少关键字个数 $\text{Min} = \lceil m/2 \rceil - 1$
- 插入关键字时，只有根节点分裂 \Rightarrow 树高增加1层
- 删除关键字时，只有根节点参与合并 \Rightarrow 树高减少1层
- 只能从根节点出发随机查找



m 阶B+树重要属性

- n 个关键字的节点只有 n 棵子树
- 上方为索引，叶子节点层存放记录
- 叶子节点层的节点通过指针相链接
- 可以从根节点出发随机查找，也可以从叶子节点层的开始节点出发顺序查找



3

哈希表查找

哈希表组成

- 存放数据的表空间，地址 $0 \sim m-1$
- 哈希函数
- 解决冲突的方法

- **哈希函数**：根据记录的关键字计算出存储地址
- **解决冲突的方法**：在出现冲突时，找另外一个存储地址。



- **开放定址法**：冲突时在周围找一个新的空闲的哈希地址。
- **拉链法**：把所有的同义词用单链表链接起来的方法。



以下关于哈希查找的叙述中错误的是（ ）。

- A. 用拉链法解决冲突易引起堆积现象 ✗
- B. 用线性探测法解决冲突易引起堆积现象
- C. 哈希函数选得好可以减少冲突现象
- D. 哈希函数 $H(k)=k \text{ MOD } p$, p 通常取小于等于表长的素数

- **同义词冲突**：两个不同关键字记录的哈希函数值相同
- **非同义词冲突**：多个不同哈希函数值的记录争抢同一地址
- **堆积现象**：指非同义词冲突出现的现象，拉链法不会引起堆积现象。