

B-树

1. 定义. m 阶 B-树:

- (1) 每个节点最多 m 棵子树.
- (2) 如果不是只有一个节点. 根节点子树 ≥ 2
- (3) 除根节点以外非叶子节点子树 $\geq \lceil \frac{m}{2} \rceil$
- (4) s 个子树 $\Rightarrow s-1$ 个关键字.

根 ≥ 2 .


非叶子 $\in [\lceil \frac{m}{2} \rceil, m]$. 子树数.

2. 插入.

关键字 $> m-1$ 则上提.

3. 删除.

(1). 底层. (删后 $\geq \lceil \frac{m}{2} \rceil - 1$). 3阶.

① 结点关键字 $> \lceil \frac{m}{2} \rceil$ 删. 

② 关键字 $\leq \lceil \frac{m}{2} \rceil$ 且剩余 + 兄弟 $\geq 2(\lceil \frac{m}{2} \rceil - 1)$ 与兄弟父结点合并分配.
 兄弟的也靠提到父结点.

③ 关键字 $\leq \lceil \frac{m}{2} \rceil$ 且剩余 + 兄弟 $< 2(\lceil \frac{m}{2} \rceil - 1)$ 与兄弟父结点一起.
 2个结点(都不满足下用3).

(2) 不在底层.

从大于该点的数据中选最小值替代被删数据.



平衡二叉树

起因:

避免出现单枝现象, 提高查找速度.

二. 概念

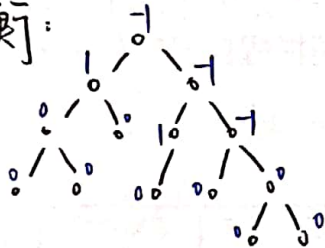
1. 平衡因子.

= 左子树高度 - 右子树高度.

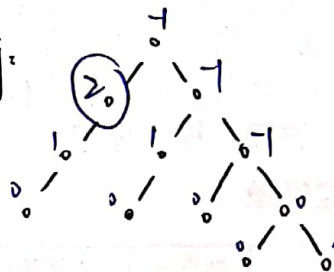
2. 平衡二叉树.

每个节点的平衡因子 $\in \{-1, 0, 1\}$

平衡:

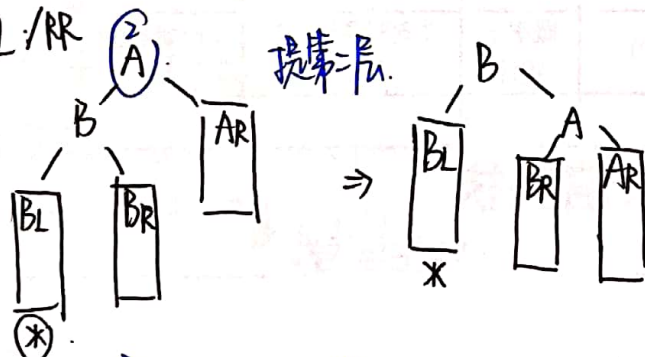


非平衡:

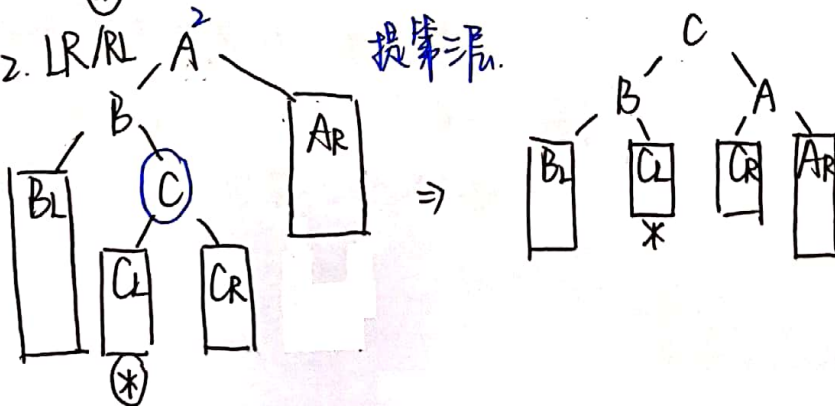


三. 再平衡法:

1. LL/RR



2. LR/RL



1. 握手定理: ~~度之和 = 2 * 边~~ $\sum d(u) = 2|E|$ 图的度和树的度不同.

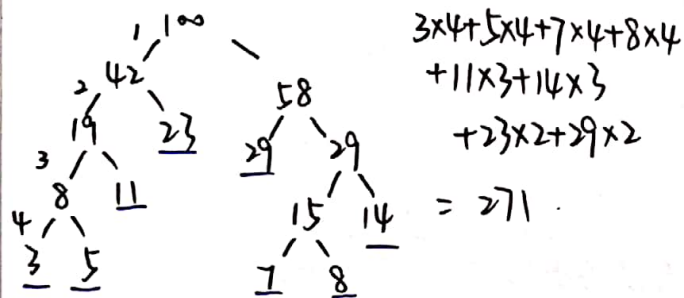
2. n 个顶点的完全图有 $\frac{n(n-1)}{2}$ 条边.

树:

1. 边 = 点 - 1. $|E| = |V| - 1 = \text{度之和}$

2. 树的度是子树的数量 (定理不可以套!)

3. Huffman 树带权路径长:



4. Huffman 树没有度为 1 的点 (n_1).

值 $\rightarrow n_0$. $\therefore n_2 = n_0 - 1$.

5. 二叉树中 $n_0 = n_2 + 1$.

6. 空树深度为 0.

7. 树的度为 m 第 i 层最多 m^{i-1} 个点.

8. 二叉树深度为 k , 则总节点数 $\in [k, 2^k - 1]$.

9. 完全二叉树: n 个节点, 深度: $\lfloor \log_2 n \rfloor + 1$.

算法:

1. 特性: 有穷、确定、可行、输入、输出.

2. 设计要求: 正确、可读、健壮、效率与存储要求.

3. 排序空间复杂度: 快排 $O(\log_2 n)$ 基数 $O(n \cdot rd)$

10. 树高 = 树深. 只是方向不同. 叶子高度 = 1.

11. 二叉树先序 = 树先序
中序 后序

12. 循环队列元素时 $front = rear$

第1-6章历年卷整合

- 在双向链表存储结构 (prior, data, next) 中, 若要在指针 p 之后插入结点 s, 则指针的操作步骤正确的是 (B)
- A. $s \rightarrow next = p \rightarrow next; s \rightarrow prior = p; p \rightarrow next = s; p \rightarrow next \rightarrow prior = s;$
 - B. $s \rightarrow next = p \rightarrow next; s \rightarrow prior = p; p \rightarrow next \rightarrow prior = s; p \rightarrow next = s;$
 - C. $s \rightarrow next = p \rightarrow next; p \rightarrow next = s; s \rightarrow prior = p; p \rightarrow next \rightarrow prior = s;$
 - D. $s \rightarrow next = p \rightarrow next; p \rightarrow next = s; p \rightarrow next \rightarrow prior = s; s \rightarrow prior = p;$

2. 下面程序段的时间复杂度为 (A)

```
Int i, j, n;
For (int i=0; i<1000; i++)
    For (j=1; j<n; j=j*2)
        S+=j*j;
```

- A. $O(\ln n)$
- B. $O(n^2)$
- C. $O(n/2)$
- D. $O(1)$

3. 二维数组 A[10][20] (下标从 0 开始) 的首地址为 2000, 每个元素占用 2 个字节。若元素按列存放, 则元素 A[6][7] 的地址为 (B)
- 行: $Location(A_{ij}) = \text{Loc}(A_{00}) + (i \cdot \text{行} + j) \times \text{字节}$
- 首地址计算: $Location(A_{ij}) = \text{Loc}(A_{00}) + (i \cdot \text{行} + j) \times \text{字节}$
- A. 2151
 - B. 2152
 - C. 2254
 - D. 2255

4. 一棵具有 2045 个结点的完全二叉树的叶子结点数为 (A)
- ② $n_0 = n_2 + 1$
 $n_1 = 0 \Rightarrow n_0 = 1023$
- A. 1023
 - B. 1024
 - C. 1025
 - D. 1026

5. 一棵 4 叉树中, 已知度为 1, 2, 3, 4 结点数分别为 6, 7, 8, 9, 则树中叶子结点数为 (B)
- ② $n_0 = n_2 + 1$
 $n_1 = 0 \Rightarrow n_0 = 1023$
- A. 50
 - B. 51
 - C. 52
 - D. 53

6. 一个有 5 棵树构成的森林, 其第一棵至第五棵树的节点数分别为 10, 8, 8, 10, 15, 由他们构造二叉树, 则对应的二叉树的左子树和右子树上的节点数分别是 (C)
- A. 10, 41
 - B. 10, 42
 - C. 9, 41
 - D. 9, 42

7. 根据使用频率为 5 个字符设计的 huffman 编码不可能的是 (C)
- A. 111, 110, 10, 00
 - B. 000, 001, 010, 011, 1
 - C. 100, 11, 10, 1, 0
 - D. 001, 000, 01, 11, 10

8. 设有 35 个值, 用它们构造哈夫曼树, 则该哈夫曼树共有 (D) 个结点
- ② $n_0 = 35$
 $n_2 = n_0 - 1 = 34$
- A. 66
 - B. 67
 - C. 68
 - D. 69

9. 一棵深度为 11 的二叉树, 最少有 (A) 个结点, 最多有 () 个结点
- A. 11, 2047
 - B. 11, 2048
 - C. 1024, 2048
 - D. 1024, 2047

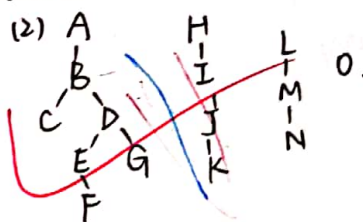
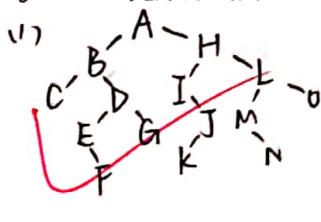
10. 已知二叉树的中序序列为 CBEFDGAIKJHMLNO, 先序序列为 ABCDEFGHIJKLMNOP

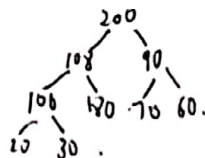
(1) 写出其对应的二叉树

(2) 写出后序序列

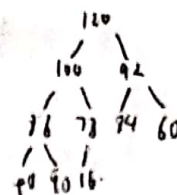
(3) 写出该二叉树对应的森林

② CFEGBDKJINMOLHA





排序



19. 以下序列为堆的是 (D)。
- A. {60, 70, 82, 91, 84, 100, 75, 98, 120} ✓
- B. {120, 100, 92, 86, 78, 84, 60, 40, 90, 16} ✓
- C. {76, 86, 92, 82, 88, 100, 160, 92, 100}
- D. {200, 108, 90, 100, 180, 70, 60, 20, 30}

20. 下面 (D) 不是一种稳定的排序方法。
- A. 选择排序 \ B. 快速排序 \ C. 希尔排序 \ D. 归并排序 稳

直接、冒泡。

不稳：简单选择、希尔、堆、快排

3. 一组记录的关键字为 {65, 55, 49, 79, 66, 35, 81, 32, 95, 56, 88, 75, 53, 20}，则利用快速排序方法并以第一个记录为支点得到第一趟排序的结果是 20, 55, 49, 53, 56, 35, 32, 65, 81, 88, 75, 66, 79, 95；第二趟排序的结果是 20, 55, 49, 53, 56, 35, 32, 65, 79, 81, 88, 75, 66, 95

4. 已知关键字的集合 {120, 86, 38, 42, 79, 66, 230, 87, 32, 73, 56, 119, 71, 53, 59, 69}，从低位到

高位采用基数排序，则经过二趟基数排序后的结果为 120, 119, 230, 38, 42, 53

56 59 62 66 69, 71 73 79 86 87

9. 在排序方法中，经过一趟排序不能确定任何元素的最终位置的排序方法有 希尔排序

____ (仅写出一种排序方法即可)。

1. 数据结构

集合/线性/树形/图.
逻辑/物理.

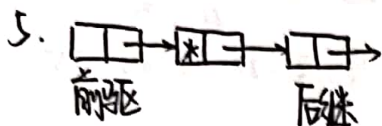
2. 数据表示

顺序映像 \rightarrow 顺序存储结构
非顺序映像 \rightarrow 链式存储结构.

3. 算法: 有穷性, 确定性, 可行, I/O.

算法设计: 正确性, 可读性, 健壮性.
效率与存储量需求.

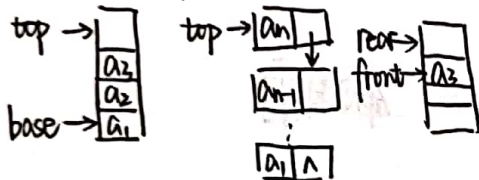
4. search 返回下标 find 返回值



6. 单链表, 静态链表, 循环, 双向.

容量有限

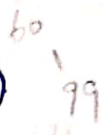
8. 顺序栈 链栈 顺序队列



循环队列

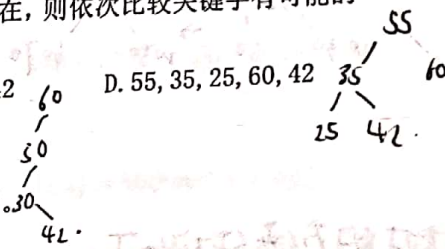


历年



- Handwritten notes showing a sequence of numbers (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20) and a diagram illustrating a sequence of numbers (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20) with arrows indicating a path or sequence.

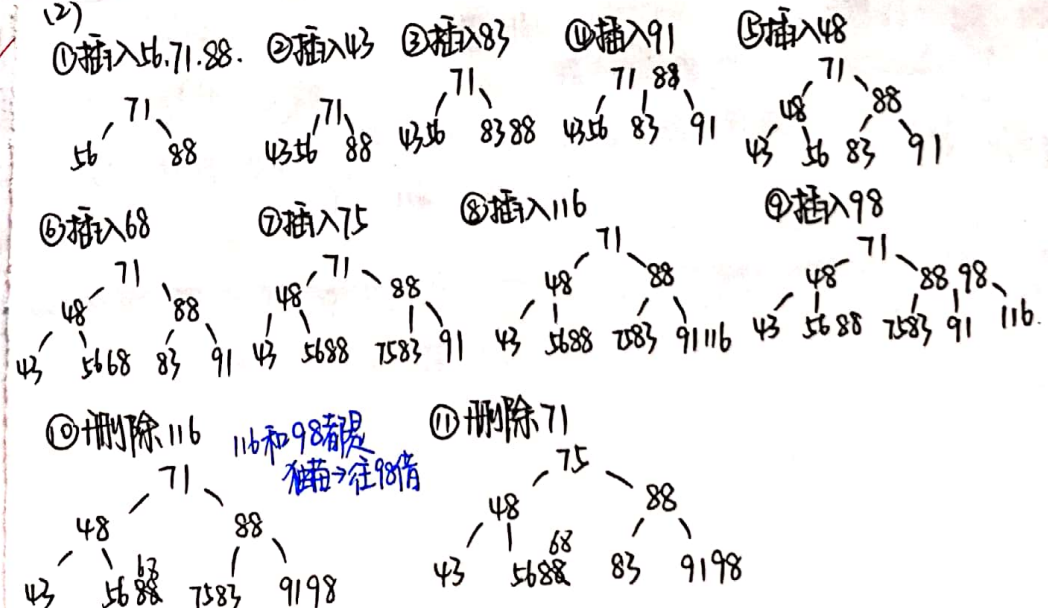
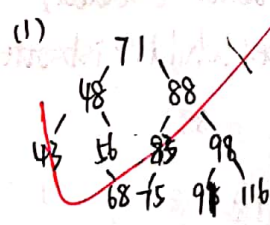
18. 在二叉排序树上查找关键字为 42 的结点, 假设该结点存在, 则依次比较关键字有可能的是 (C)。
- A. 60, 72, 50, 42 B. 30, 20, 50, 42 C. 60, 50, 30, 42 D. 55, 35, 25, 60, 42



8. 对二叉排序树采用 中序 遍历可以得到一个递增序列。 30、42

3. 设关键字输入序列为 $\{56, 71, 88, 43, 83, 91, 48, 68, 75, 116, 98\}$

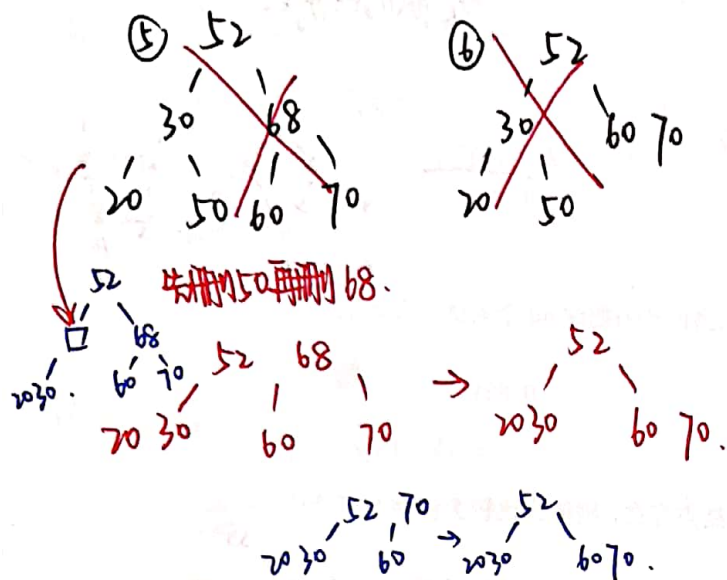
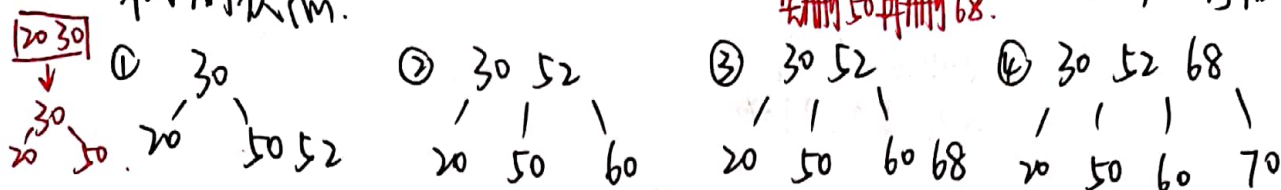
- (1) 试构造平衡二叉树。
(2) 构造 3 阶 B 树, 并分别写出依次删除 116 和 71 后的 B 树。
(3) HASH 表表长为 16, HASH 函数为 $H(\text{key}) = \text{key} \% 13$, 试用二次探测再散列解决冲突的方法构造哈希表。



- (3) 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
91 68 56 43 71 98 48 83 88 75 116

试从空树开始, 画出按以下次序向 2-3 树中插入数据的建树过程
20, 30, 50, 52, 60, 68, 70. 如果以后删除 50 和 68, 画出每一步执行后 2-3 树的状态.

选择题
1. 数据结构
A. 操作对象
B. 在数据



第一章 (作业)

选择题

数据结构是一门研究非数值计算的程序设计问题中计算机的(A)、以及它们之间的(C)和运算的学科。

- A. 操作对象 B. 数据映象 C. 关系 D. 算法

2. 在数据结构中, 逻辑上数据结构可分为()。

- B. 动态结构和静态结构 B. 线性结构和非线性结构
C. 紧凑结构和非紧凑结构 D. 内部结构和外部结构

3. 算法分析考虑()两方面的问题。

- D. A. 正确性和空间复杂性 B. 易读性和健壮性
C. 数据复杂性和程序复杂性 D. 时间复杂性和空间复杂性

4. 算法分析的目的是()。

- B. A. 找出数据结构的合理性 B. 分析算法的效率;
C. 研究算法中输入和输出关系 D. 分析算法的易理解性

5. 下面()是算法设计的要求。

- A. A. 正确性 B. 确定性
C. 输入、输出 D. 有穷性

二、填空题

1. 算法的5个特性为有穷性、确定性、输入、输出和正确性、可行性
2. 顺序存储结构是用一组地址连续的空间存放数据元素, 逻辑上相邻的数据元素, 物理上一定相邻。链式存储结构是用一组地址任意的空间存放数据元素, 逻辑上相邻的数据元素, 物理上不一定相邻。
3. 数据结构在计算机中的表示, 包括数据结构中数据元素和数据元素之间关系的表示。
4. 算法效率的度量方法有时间复杂度和空间复杂度 事前分析、事后统计
5. 一个没有循环的算法中的基本运算次数与问题规模n无关, 其时间复杂度记为 $O(1)$ 。

三、简答题

1. 数据结构有哪四种形式? 各有什么特点? 线性、图、树、栈
2. 数据结构、数据类型和抽象数据类型有什么区别?
3. 设有以下3个函数:

$$f(n) = 21n^4 + 2n^2 + 1000, \quad g(n) = 15n^4 + 500n^2, \quad h(n) = 5000n^{3.5} + n \log_2^n$$

请判断下列断言是否正确:

- (1) $f(n)$ 是 $O(g(n))$ \checkmark
(2) $h(n)$ 是 $O(f(n))$ \times
(3) $g(n)$ 是 $O(h(n))$ \times
(4) $h(n)$ 是 $O(n^{3.5})$ \checkmark
(5) $h(n)$ 是 $O(n \log_2^n)$ 错误 \times

三、计算下列算法的时间复杂度。

```
1. int fun1(int n){
    int s=0, i=0;
    while(s<n){
        s+=i;
        i++;
    }
}
```

$s = 0 \quad i = 1$
 $s = 1 \quad i = 2$
 $s = 3 \quad i = 3$
 $s = 6 \quad i = 4$
 $s = 10 \quad i = 5$

$O(\log n)$
 $O(\sqrt{n})$

```
2. void fun3(){
    int x=91, y=100;
    while(y>0){
        if(x>100){
            x-=10;
            y--;
        }
        else x++;
    }
}
```

$O(1)$

$x = 91 \quad y = 100 \quad 1$
 $x = 93 \quad y = 100 \quad 2$
 $x = 94 \quad y = 100 \quad 3$
 $x = 100 \quad y = 100 \quad 9$
 $x = 101 \quad y = 100 \quad 10$
 $x = 91 \quad y = 99$

四、算法设计

构造顺序表的到置算法(空间复杂度最小的情形)

查找 (作业)

一、选择题

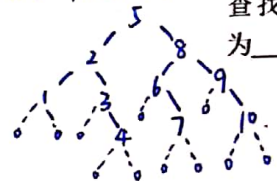
1. 有 20 个元素的按值有序的顺序表, 采用折半查找第 6 个元素需要经过 (C) 次比较。
A. 2 B. 3 C. 4 D. 5
2. 长度为 20 的有序表采用折半查找, 共有 () 个元素查找长度为 4, 共有 (C) 个元素查找长度为 5。
A. 8, 16 B. 7, 16 C. 8, 5 D. 7, 5
3. 深度为 7 的平衡二叉树最小有 (D) 结点。有个公式: $a[i] = a[i-1] + a[i-2] + 1$
A. 7 B. 12 C. 20 D. 33
4. 已知关键字的集合 {70, 39, 41, 101, 86, 33, 26, 130, 75, 53, 60}, 构造二叉排序树, 该二叉排序树的树高为 (B)。
A. 4 B. 5 C. 6 D. 7
5. 由关键字的集合 {60, 40, 100, 80, 78, ...} 构造平衡二叉树, 当插入 78 时引起不平衡, 则其旋转型为 (A)。
A. LL B. LR C. RL D. RR
6. 在二叉排序树上查找关键字为 45 的结点, 假设该结点存在, 则依次比较关键字有可能的是 (B)。
A. 60, 35, 40, 45 B. 40, 70, 48, 45 C. 20, 79, 40, 45 D. 45, 25, 45, 50, 45
7. 采用链地址法解决冲突时, 每个散列地址所链接的同义词子表各个表项的 (C) 相同。
A. 关键字值 B. 数据元素值 C. 散列地址 D. 含义
8. 采用线性探测再散列解决冲突时产生的一系列后继散列地址 (B)。
A. 必须大于等于原散列地址 B. 可以大于或小于但不等于原散列地址 C. 必须小于等于原散列地址 D. 对地址在何处没有限制
9. 散列法存储的基本思想是根据 (C) 来决定存储地址。
A. 散列表空间 B. 元素的序号 C. 关键字值 D. 装填因子
10. 散列地址空间为 m , 关键字为 key , 哈希函数为 $H(key) = key \bmod p$, 为了减少冲突的频率, 一般 p 为 (B)。
A. 小于 m 的最大奇数 B. 小于 m 的最大素数 C. 小于 m 的最大合数 D. 大于 m 的最小素数
11. 在一棵 50 阶 B-树中删除一个结点引起该结点与左兄弟结点的合并, 则其左兄弟上的关键字个数为 (B)。
A. 23 B. 24 C. 25 D. 26
12. 假定有 40 个关键字值互为同义词, 若采用线性探测再散列把这 40 个关键字值存入散列表中, 至少要进行 (B) 次探查。
A. 821 B. 820 C. 819 D. 818
13. 下面关于散列表的说法正确的是 ()。
A. 散列函数构造的越复杂越好, 因为这样随机性好, 冲突小
B. 除留余数法是所有散列函数中最好的
C. 在散列表中删除一个元素, 不管用何种方法解决冲突都只要简单的将该元素删去即可
D. 不存在特别好与坏的散列函数, 要视情况而定

二、填空题

1. 动态查找表与静态查找表的重要区别在于前者包含 插入 和 删除 运算, 而后者不包含这两种运算。

2. 长度为 30 的顺序表采用顺序存储结构存储, 并采用折半查找技术, 在等概率的情况下, 查找成功时的平均查找长度为 3.5。查找不成功时的平均查找长度为 3.5。

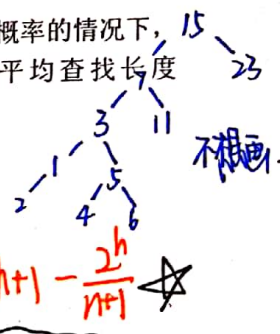
例: 一个长为 10 的表:



度为 1 的结点 → 还有一个 (例: 结点 4)
0 → 2 (例: 结点 4)

也可以是完全二叉树一棵
长为 30: 1×1
 2×2
 4×3
 8×4 还有 1 个
 15×5 还有 15 个

$\frac{1 \times 4 + 30 \times 5}{31} \rightarrow$ 路径和
 \rightarrow 所有结点的数量



$h+1 - \frac{2^h}{n+1}$ ★

查找(作业)

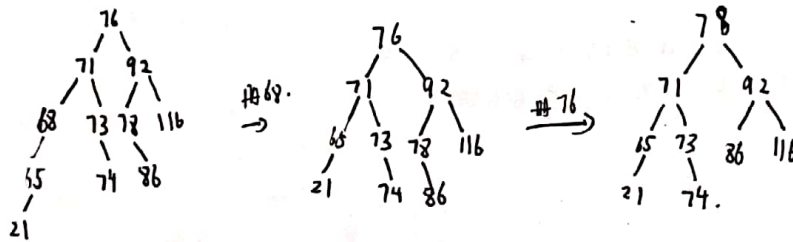
3. 长度为 60 的有序表采用折半查找, 共有 29 个元素的查找长度为 6。
4. 折半查找的要求是 顺序存储 和 有序排列。
5. 一棵二叉排序树按 中序 遍历可以得到有序序列。
6. 若一棵 5 阶 B-树的高度是 5 (叶子层不算), 则这棵 B-树至少有 352 个关键字。至多有 5621 个关键字。
7. 已知一棵 5 阶 B-树共有 58 个关键字, 则该 B-树的最大高度为 4; 该 B-树的最小高度为 3。
8. 设散列表长度为 14, 散列函数 $H(key) = key \bmod 11$, 表中已有 4 个元素 15, 38, 61, 84, 其余地址为空, 此散列表采用 二次探测再散列 解决冲突, 现需插入新元素 49, 则 49 的存储位置是 8。

4 → 15
5 → 38
6 → 61
7 → 84

三、问答题

1. 设关键字输入序列为依据关键字序列 {76, 71, 92, 68, 73, 78, 86, 74, 65, 116, 21}

- (1) 构造二叉排序树, 并分别写出删除 68 和 92 后的二叉排序树;
- (2) 构造平衡二叉树;
- (3) 构造 3 阶 B-树, 并分别写出依次删除 74 和 76 后的 B-树。
- (4) HASH 表表长为 16, HASH 函数为 $H(key) = key \% 13$, 试用二次探测再散列解决冲突的方法构造哈希表。



线性表 (作业)

选择题

1. 线性表是具有 n 个 C 的有限序列。
A. 表元素 B. 字符
C. 数据元素 D. 数据项

2. 线性表采用链表存储时, 其地址 D。
A. 必须是连续的 B. 一定是不连续的
C. 部分地址必须是连续的 D. 连续与否均可以

3. 线性表的静态链表存储结构与顺序存储结构相比优点是 D。
A. 所有的操作算法实现简单 B. 便于随机存取
C. 便于插入和删除 D. 便于利用零散的存储器空间

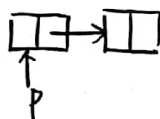
4. 设线性表有 n 个数据元素, 以下操作中, A 在顺序表上实现比在链表上实现效率更高。
A. 输出第 i ($1 \leq i \leq n$) 个数据元素值
B. 交换第 1 个数据元素与第 2 个数据元素的值
C. 顺序输出这 n 个数据元素的值
D. 输出与给定值 x 相等的元素在表中的符号

5. 设线性表中有 $2n$ 个数据元素, 以下操作中, A 在单链表中实现要比在顺序表中实现效率更高。
A. 删除指定的数据元素 *虽然都是 $O(n)$*
B. 在最后一个数据元素的后面插入一个新的数据元素
C. 顺序输出前 K 个数据元素
D. 交换第 i 个数据元素和第 $2n - i + 1$ 个数据元素的值 ($i = 0, 1, K, n - 1$)

6. 如果最常用的操作是取第 i 个结点及其前驱, 则采用 D 存储方式最节省时间。
A. 单链表 B. 双链表
C. 单循环链表 D. 顺序表

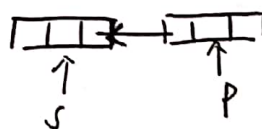
7. 与单链表相比, 双链表的优点之一是 D。
A. 插入、删除操作更简单 B. 可以进行随机访问
C. 可以省略表头指针或表尾指针 D. 访问前后相邻结点更灵活

8. 在一个单链表中, 若删除 p 所指结点的后继结点, 则执行 A。
A. $p \rightarrow next = p \rightarrow next \rightarrow next;$
B. $p \rightarrow next = p \rightarrow next;$
C. $p = p \rightarrow next \rightarrow next;$
D. $p = p \rightarrow next; p \rightarrow next = p \rightarrow next \rightarrow next;$



9. 在双向链表存储结构中, 若要删除指针 p 所指的结点的前驱结点 (若存在), 则指针的操作步骤正确的是 C。

- A. $s = p \rightarrow prior; p \rightarrow prior \rightarrow next = p \rightarrow next;$
 $p \rightarrow next \rightarrow prior = p \rightarrow prior; delete s;$
B. $s = p \rightarrow prior; p \rightarrow prior = p \rightarrow prior \rightarrow next;$
 $p \rightarrow prior \rightarrow prior \rightarrow next = p; delete s;$
C. $s = p \rightarrow prior; p \rightarrow prior \rightarrow prior \rightarrow next = p;$
 $p \rightarrow prior = p \rightarrow prior \rightarrow prior; delete s;$
D. $s = p \rightarrow prior; p \rightarrow next \rightarrow next \rightarrow prior = p;$



线性表2 (作业).

$p \rightarrow next = p \rightarrow next \rightarrow next; delete\ s;$

10. 线性表中最常用的操作是在最后一个数据元素之后插入一个数据元素和删除第一个数据元素, 则采用 双链表 存储方式最节省运算时间。

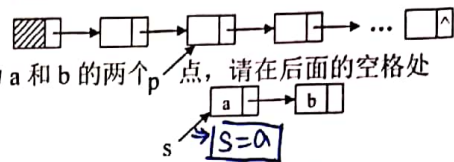
- A. 单链表
C. 双链表

- B. 仅有头指针的单循环链表
D. 仅有尾指针的单循环链表

二、填空题

1. 在如下图所示的链表中, 若在指针p所指的结点之后插入数据域值相继为a和b的两个点, 请在后面的空格处填入正确的语句以实现该操作。

$p \rightarrow next = p \rightarrow next;$
 $s \rightarrow next \rightarrow next = p \rightarrow next;$



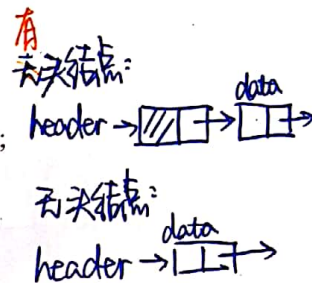
2. 若线性表的数据元素总数基本稳定, 很少进行插入和删除, 但要求以最快的速度存取表中的数据元素, 这时应采用 顺序表 存储表示。若经常进行插入删除操作, 则该线性表应采用 链式 存储表示。

3. 已知L是无表头结点的单链表, 且P结点既不是首元结点, 也不是尾结点, 试从下列提供答案中选择合适的语句序列:

- a) 在P结点后插入S结点的语句序列是 41;
b) 在P结点前插入S结点的语句序列是 711841;
c) 在表首插入S结点的语句序列是 11513512;
d) 在表尾插入S结点的语句序列是 916961;

- (1) $P \rightarrow next = S;$
(2) $P \rightarrow next = P \rightarrow next \rightarrow next;$
(3) $P \rightarrow next = S \rightarrow next;$
(4) $S \rightarrow next = P \rightarrow next;$
(5) $S \rightarrow next = L;$
(6) $S \rightarrow next = NULL;$
(7) $Q = P;$

- (8) $while(P \rightarrow next \neq Q) P = P \rightarrow next;$
(9) $while(P \rightarrow next \neq NULL) P = P \rightarrow next;$
(10) $P = Q;$
(11) $P = L;$
(12) $L = S;$
(13) $L = P;$



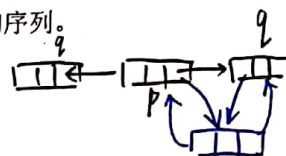
4. 已知p结点是某双向链表的中间结点, 试从下列提供的语句中选择合适的语句序列。

- a) 在p结点后插入s结点的语句序列是 1276312736;
b) 在p结点前插入s结点的语句序列是 85458134;
c) 删除p结点的直接后继结点的语句序列是 151118;
d) 删除p结点的直接前驱结点的语句序列是 1621018;
e) 删除p结点的语句序列是 91417;

- (1) $p \rightarrow next = p \rightarrow next \rightarrow next;$
(2) $p \rightarrow prior = p \rightarrow prior \rightarrow prior;$
(3) $p \rightarrow next = s;$
(4) $p \rightarrow prior = s;$
(5) $s \rightarrow next = p;$
(6) $s \rightarrow prior = p;$
(7) $s \rightarrow next = p \rightarrow next;$
(8) $s \rightarrow prior = p \rightarrow prior;$
(9) $p \rightarrow prior \rightarrow next = p \rightarrow next;$

- (10) $p \rightarrow prior \rightarrow next = p;$
(11) $p \rightarrow next \rightarrow prior = p;$
(12) $p \rightarrow next \rightarrow prior = s;$
(13) $p \rightarrow prior \rightarrow next = s;$
(14) $p \rightarrow next \rightarrow prior = p \rightarrow prior;$
(15) $q = p \rightarrow next;$
(16) $q = p \rightarrow prior;$
(17) $delete\ p;$
(18) $delete\ q;$

$s \rightarrow next = p$ ⑤
 $p \rightarrow prior \rightarrow next = s$ ⑬
 $s \rightarrow prior = p \rightarrow prior$ ⑧
 $p \rightarrow prior = s$



三、算法设计

1. 已知一个带头结点的单链表ha中存放一组整型数, 构造一算法将链表ha中值为偶数的结点加入链表hb中, 且链表hb按值非递减排列。函数原型为

`void Inserthb(LNode *ha, LNode *&hb).`

2. 已知2个带头结点的单链表ha和hb按值非递减存储一组整型数, 构造一算法计算 $ha = ha \cup hb$, 计算后链表仍然有序。并返回并集链表中元素个数。