考试科目名称 数据结构 （期中考试）

2018——2019 学年第 一 学期 教师 姜远、商琳、刘佳 考试方式：闭卷

系（专业） 计算机 年级 2017 班级 [redacted]

学号 [redacted] 姓名 [redacted] 成绩 _____

题号	一	二	三	四	总分
分数	10	18	24	18	80

得分 [ ]

一、单选题（每小题 2 分，本题满分 20 分）

1. 设一个线性表最常用的操作是查找指定序号的元素和在末尾插入元素，则选用（ ）结构最节省时间。

A. 顺序表 B. 单链表 C. 带头结点的双循环链表 D. 带尾指针的单循环链表

2. 一棵左子树为空的二叉树在先序线索化后，其中空的链域的个数是（ ）。

A. 不确定 B. 0 C. 1 D. 2

3. 一棵二叉树的先序序列和后序序列相反，则该二叉树一定满足：（ ）

A. 其中只有一个叶子结点 B. 其中任意结点没有左孩子

C. 其中任意结点没有右孩子 D. B 或 C

4. 若从二叉树的任一结点出发到根的路径上所经过的结点序列按其关键码有序，则该二叉树是（ ）。

A. 中序遍历有序 B. 哈夫曼树 C. 堆 D. 完全二叉树

5. 在一个有 125 个元素的顺序表中插入一个新元素并保持原来顺序不变，平均要移动（ ）个元素。

A. 8 B. 62.5 C. 62 D. 7

6. 完全二叉树的某结点若无左子女，则它必是（ ）。

A. 倒数第二层的最后一个结点 B. 最后层的最后一个结点

C. 叶结点 D. 最后层的第一个结点

7. 一个栈的入栈序列是：a,b,c,d,e，则栈不可能的输出序列是（ ）。

A. edcba B. decba C. dceab D. abcde

8. 在下述结论中，正确的是（ ）。

①只有一个结点的二叉树的度为 0； ②二叉树中结点的度为 0 或 2；

③二叉树的左右子树可任意交换；

④深度为 K 的完全二叉树的结点个数小于或等于深度相同的满二叉树。

A．①②③　　B．②③④　　C．②④　　D．①④

9. 森林 F 对应的二叉树为 B，它有 m 个结点，B 的根为 p，p 的右子树结点个数为 n，森林 F 中第一棵树的结点个数是（　　）。

A. m-n　B．m-n-1　C．n+1　D．条件不足，无法确定

10. 二叉树在线索后，仍不能有效求解的问题是（　　）。

A．前（先）序线索二叉树中求前（先）序下的后继

B．中序线索二叉树中求中序下的后继

C．中序线索二叉树中求中序下的前驱

D．后序线索二叉树中求后序下的后继

得分　　**二、填空题（每空 2 分，本题满分 26 分）**

1. 循环队列 QU 存储于数组中（数组空间为 K），front 指向队头元素，rear 指向队尾元素之后的空位置，队列中的元素个数是 _____。

2. 中缀表达式 A*(B+C)-D 的后缀形式为：_____。

3. 深度为 k（设根结点为 1 层）的二叉树上，只有度为 0 和度为 2 的结点，则这类二叉树上所含结点总数最少为 _____ 个，至多为 _____ 个。

4. 设 W 为一个二维数组，其每个数据元素占用 6 个字节，行下标 i 从 0 到 8，列下标 j 从 0 到 3。W 中第 6 行的元素和第 4 列的元素共占用 _____ 个字节。若按行主顺序存放二维数组 W，其起始地址为 100，则二维数组 W 的最后一个数据元素的起始地址为 _____。

5. 广义表 A=((a,b),(),((c,d),e))的长度为 _____，深度为 _____。

6. 设输入序列为 a、b、c、d、e、f，则经过入栈和出栈的组合后可以得到 _____ 种不同的输出序列。

7. 已知广义表 A=((a,b,c),(d,e,f),(h,(i,j)),g)，通过 head 和 tail 运算从 A 表中取出原子项 e 的运算是 _____。

8. 以二叉链表作为二叉树存储结构，对有 K 个结点的二叉链表进行线索化时，可以作为线索的链域的个数为 _____。

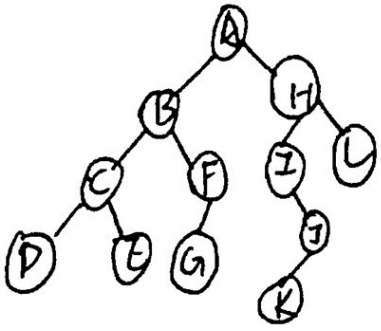
9. 假定 front 和 rear 分别为一个带表头结点的链式队列的队头和队尾指针，则该链式队列中队列为空和只有一个结点的条件是 _____ 和 _____。(假设结点的指针域为 link)

得分 | | 三、解答题（每小题 6 分，本题满分 36 分）

1. 已知一棵二叉树的先序、中序遍历序列如下，画出该二叉树，并给出其后序遍历序列。

先序：A B C D E F G H I J K L

中序：D C E B G F A I K J H L



后序(LRV) DECGFBKJILHA

2. 已知四个字符 S，T，U，Y 的哈夫曼编码分别是 1,01,000,001，下列的 0、1 串是由以上 4 个字母构成的一段文本的哈夫曼编码。

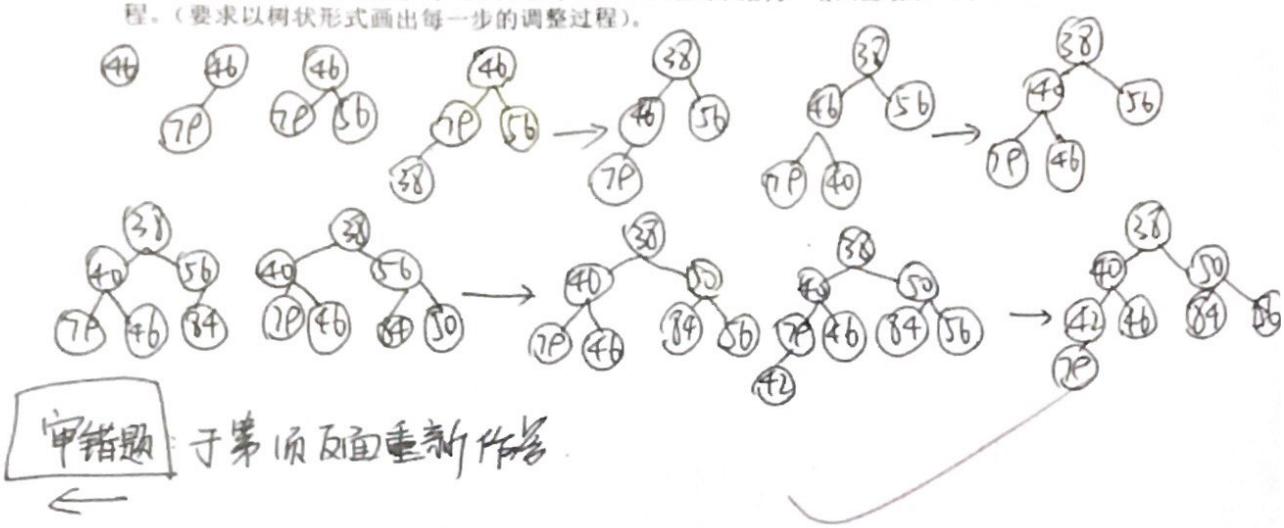
S T U Y

100100001101101001101001

请将上述 0、1 串还原为编码前的文本，并画出字符 S，T，U，Y 构成的哈夫曼树。

编码：SUTSTSTYSTYS

3. 已知一组记录为(46,79,56,38,40,84,50,42)，将其整理成堆结构，请画出最小堆的构造过程。（要求以树状形式画出每一步的调整过程）。



审错题：于第1页反面重新作答

4. 设二叉树 T 的存储结构如下：

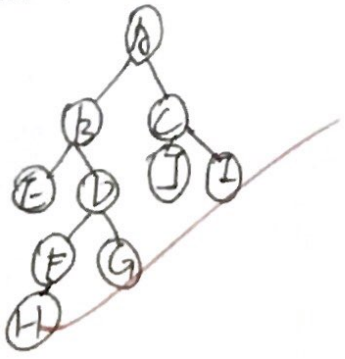
	0	1	2	3	4	5	6	7	8	9
Lchild	-1	-1	1	2	7	4	0	-1	-1	-1
Data	J	H	F	D	B	A	C	E	G	I
Rchild	-1	-1	-1	8	3	6	9	-1	-1	-1

其中 Lchild,Rchild 分别为结点的左、右孩子指针域，Data 为结点的数据域。

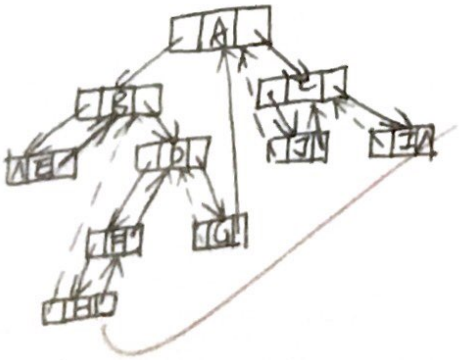
(1) 画出二叉树 T 的逻辑结构；

(2) 画出二叉树的中序线索化二叉树。

(1)

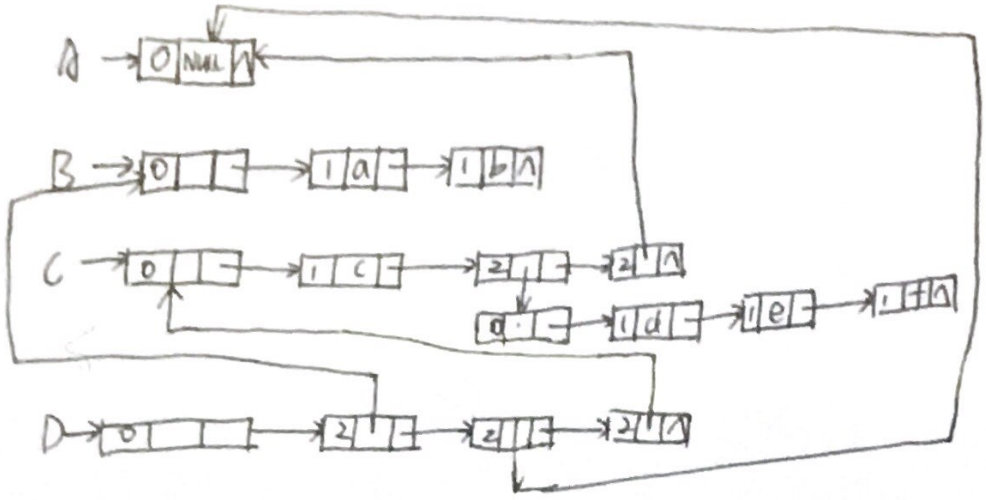


(2)



2 得分

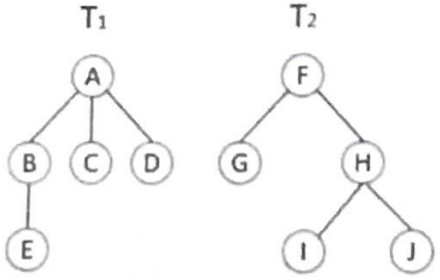
5. 给定四个广义表 A=()，B=(a, b)，C=(c, (d, e, f), A)，D=(B, A, C)。画出这四个广义表的链表存储表示，其中链表结点包含(标志域，信息域，尾指针域)。

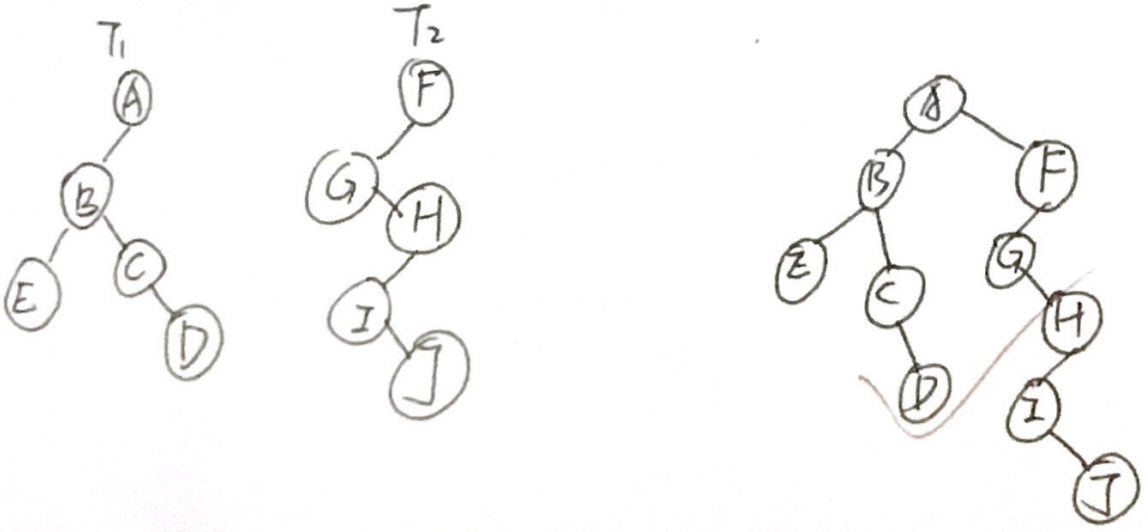


引用

-2

6. 给定下图所示的森林 F={T1,T2}，基于子女-兄弟表示，先将T1与T2分别转化为二叉树，再将F转化为二叉树。





得分 18 四、 算法题 (本题满分 18 分)

1. 算法 F 及所引用的数组 A 的值如下，写出调用 F(1)的运行结果，其中 n=15。(9 分)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	B	C	D	E	F	G	O	O	H	O	I	J	K	L

```
void F(int i) {
    if ( (i <=n) && (A[i] != 'O') ) {
        F(2* i) ;
        cout<<A[i] ;
        F(2* i +1) ;
    }
}
```

F

2. 假设有两个按元素值递增次序排列的线性表，均以单链表形式存储。请编写算法将这两个单链表合并为一个按元素值递减次序排列的单链表，并要求利用原来两个单链表的结点存放合并后的单链表，其中 la, lb 分别为两个单链表的头指针，链表节点 Linknode 包含（data, link）两个域: data 存放数据，link 指向后继节点。(9 分)

```
LinkNode* Merge(LinkNode* la, LinkNode* lb) {
    LinkNode * ta =la ;
    LinkNode * tb =lb;
    LinkNode *head= NULL;
    LinkNode * tail=NULL;
    while (ta &&tb){
        LinkNode *add ;
        if(ta->data >tb-> data){
            add = ta;
            ta= ta-> link
        } else{
            add = tb;
            tb= tb -> link;
        }
        if(head){
            tail->link =add;
            tail = add;
        }else{
            head =tail =add;
        }
    }
    if(tail){
        while(ta){
            tail->link =ta;
            tail= tail->link;
            ta= ta->link;
        }while(tb){
            tail->link =tb;
            tail= tail->link;
            tb= tb->link;
        }
    }else{
        tail = ta ? ta : tb;
    }
    tail -> link=NULL;
    Reverse (head);
    return head;
}
void Reverse (LinkNode* head){
    LinkNode * i = head;
    while(i&&i->link){
        LinkNode *j = i->link;
        LinkNode *min = i;
        while (j){
            if (min-> data > j->data)
                min=j;
        }
        if (min != i) {
            int data = i->data;
            i -> data = min->data;
            min->data = data;
        }
    }
}
```