**2019年全国硕士研究生统一入学考试自命题试题（A卷）**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

招生专业与代码：计算机科学与技术、软件工程、网络空间安全、工程硕士

研究方向：计算机系统结构081201，计算机软件与理论081202，计算机应用技术081203，软件工程083500，计算机技术(专业学位) 085211，网络空间安全083900  
考试科目名称及代码：数据结构830

**一、 单项选择题(每题2分，共30分)**

1. 在任意一棵二叉树的先序序列和后序序列中，各叶子之间的相对次序关系 ( **D** 　)。

A．不一定相同 B．互为逆序 C．都不相同 D．都相同

2. 深度为4的二叉树至多有结点数为（　**C** 　）。

A. 18 B. 14 C. 15 D.16

3. 在一个具有n个顶点的有向图中，若所有顶点的入度数之和为m，则所有顶点的度数之和为（ **D**  ）。

A．m B．m-1 C．m+1 D．2m

4. 快速排序在( **D** )情况下最不利于发挥其长处。

A. 被排序的数据量太大. B. 被排序数据中含有多个相同的关键字

C. 被排序的数据完全无序 D. 被排序的数据已基本有序

5. 一组记录的关键字为（45,80,55,40,42,85）, 则利用堆排序的方法建立的初始堆为（ **B** ）。

A. （80,45,55,40,42,85) B. (85,80,55,40,42,45)

C. (85,80,55,45,42,40) D. (85,55,80,42,45,40)

6. 对有18个元素的有序表（下标为1~18）作折半查找,则查找A[3]的比较序列的下标为 ( **D** )。

A. 1,2,3 B. 9,5,2,3 C. 9,5,3 D. 9,4,2,3

7. 具有n个顶点的完全有向图的边数为( **B**  )。

A. n(n-1)/2 B. n(n-1) C. n2 D. n2-1

8. 利用逐点插入法建立序列（50,72,43,85,75,20,35,45,65,30）对应的二叉排序树以后，查找元素35要进行（ **A** ）。

A. 4次 B. 5次 C. 3次 D. 2次

9. 判定一个有向图是否存在回路除了可以利用拓扑排序方法外，还可以用（ **D** ）。

A．求最短路径的Floyd方法 B．求最短路径的Dijkstra方法

C．广度优先遍历算法 D．深度优先遍历算法

10. 对于一个具有n个顶点的无向连通图，它包含的连通分量的个数为（  **B** ）。

A．0 B．1 C．n D．n+1

1. 在一个单链表中,若p所指的结点不是最后一个结点,在p之后插入s所指的结点, 则执行(　**D** 　)。

A. s->next=p; p->next=s B. p->next=s; s->next=p

C. p=s; s->next=p->next D. s->next=p->next; p->next=s

1. 设F是由T1、T2和T3三棵树组成的森林，与F对应的二叉树为B，T1、T2和T3的结点数分别为N1、N2和N3，则二叉树B的根结点的左子树的结点数为（ **A**  ）。

A. N1-1 B. N2-1 C. N2+N3 D. N1+N3

1. 设输入元素1,2,3,P,A，输入次序为：123PA，元素经过栈后到达输出序列。当所有元素均达到输出序列，下面（ **B** ）序列可以作为高级语言的变量名。
2. 123PA B. PA321 C. 12AP3 D. PA123
3. 在一个链队列Q中，删除一个结点需要执行的指令是（ **C**  ）。

A. Q.rear=Q.front->next; B. Q.rear->next=Q.rear->next->next;

C. Q.front->next=Q.front->next->next; D. Q.front=Q.rear->next;

15. 如果T2是由树T转换而来的二叉树,那T中结点的后序就是T2中结点的 ( **B**  )。

1. 先序 B. 中序 C. 后序 D. 层次序

**二．填空题(每空2分，共20分)**

1. 设根结点在第一层，那么具有n个结点的完全二叉树，其高度为 。

2. 对于一个循环队列Q[0..m-1]，队头、队尾指针分别为f、r，其判空的条件是 **f==r**  ，判满的条件是  **f==(r+1)%m**  。

3. 在堆排序，希尔排序，快速排序，归并排序算法中，占用辅助空间最多的是 **归并排序**  。

4. 已知二维数组A[m][n]采用行序为主序存储，每个元素占k个存储单元，并且第一个元素的存储地址是Loc(A[0][0]), 则A[i][j]的地址是**Loc(A[0][0])+k\*(i\*n+j)**  。

5. 若某记录序列的关键字序列是（235，346，021，558，256），用链式基数排序方法排序，第一次收集的结果是  **(021,235,346,256,558)**  。

1. 设Hash表为m=11，散列函数H(k)=k%11，表中已有4个结点，地址分别为：addr(15)=4,addr(38)=5,addr(61)=6,addr(84)=7，其余地址为空。如果用二次探测再散列处理冲突，关键字为49的结点的地址是  **9**  。
2. 在一个3阶的B-树上，每个结点包含的子树相同，最多为 个结点，最少为
3. 个结点。
4. 一个连通图的 **生成树**  是一个极小连通子图。

**三．判断题（每题1分，共10分，正确的选t，错误的选f）**

1. 对于n个记录的集合进行冒泡排序，在最坏情况下的时间复杂度是O(n2)。 ( **√**  )
2. 包含两个结点的所有二叉树都是相同的。（ **×** ）
3. 一个图按广度优先遍历的结果是唯一的。( × )
4. 用Prime算法和Kruskal 算法求得的图的最小生成树一定相同。（ × ）
5. 线性表中的每一个元素都有一个前驱和后继元素。( × )
6. 在n个顶点的无向图中，若边数>n-1，则该图必是连通图。（ × )
7. 完全二叉树的某结点若无左孩子，则必是叶子结点。 （ **√**  )
8. 在B-树,有n棵子树的结点中有n个关键字。 （ × ）
9. 在一个有向图的邻接表或逆邻接表中，如果某个顶点的链表为空，则该顶点的度一定为零。（ × )
10. 用邻接矩阵法存储一个图时，在不考虑压缩存储的情况下，所占用的存储空间大小与图中结点的个数有关，而与图的边数无关。（ **√**  ）

**四. 简答题（40分）**

1. 设G为有n个顶点的无向连通图，证明G至少有n-1 条边。 （7分）

**答：假设使用n-1条无向边可以使得含有n个顶点得无向图连通。**

**①当n=1时，成立**

**②设当n=k时，使用k-1条无向边可以使得无向图连通**

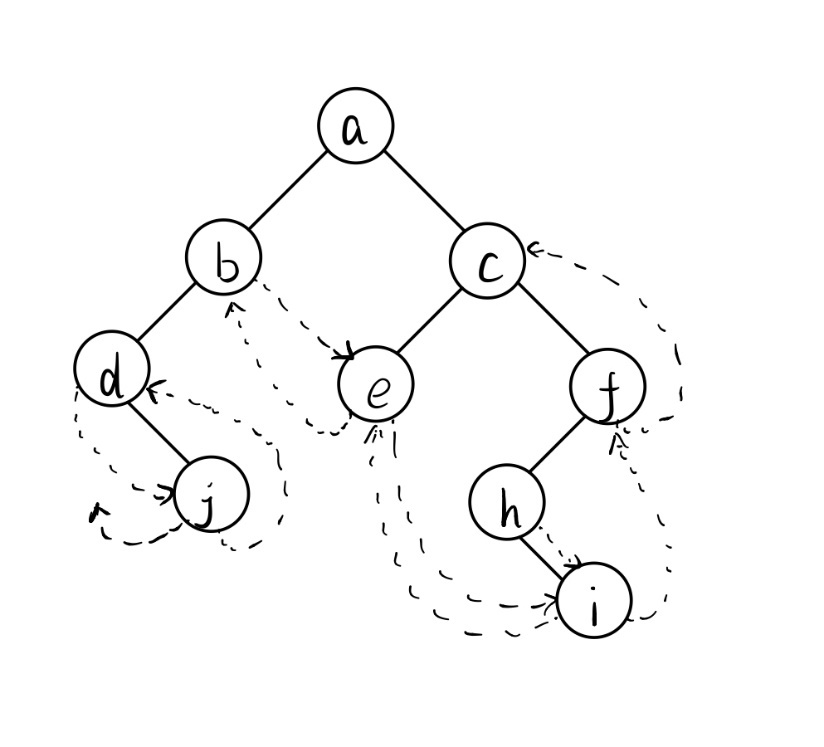
**③当n=k+1时，因为k-1条无向边可以使得k个顶点连通，则无向图G有k+1个顶点，使用k-1条无向边将k个顶点连通后，无向图G有两个连通分量，从两个连通分量中各取一个顶点建立一条无向边则使得整个图连通，即使用k条无向边可以使得含有n个顶点的无向图连通，成立。**

**又因为一条边仅可以连接两个顶点，故当无向图G的顶点数为n，显然n-2条无向边无法使得无向图连通。**

**由此可得，G为有n个顶点的无向连通图至少有n-1 条边证明成立。**

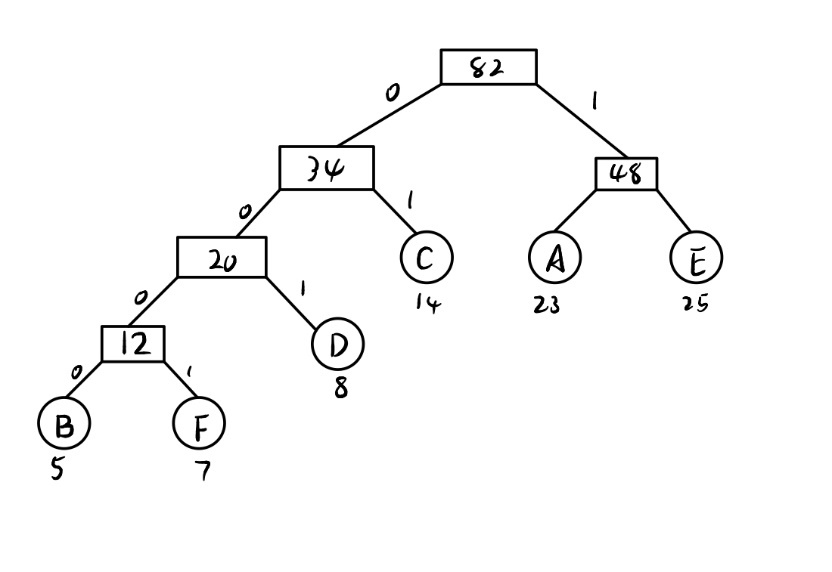
1. 什么是线索二叉树？一棵二叉树的中序遍历序列为djbaechif，前序遍历序列为abdjcefhi,请画出该二叉树的后序线索二叉树。（7分）

答：



**解析：规定若无左子树则令lchild指向其前驱结点；若无右子树，则令rchild指向其后继结点，还需要增加两个标志域标识指针域是指向左右孩子还是指向前驱后继，以这种结构构成的二叉链表作为二叉树的存储结构，成为线索链表，加上线索的二叉树称为线索二叉树。**

1. 已知某通讯电文仅有A、B、C、D、E、F六个字符构成，其出现频率分别为23,5,14,8,25,7,请给出他们的Huffman编码以及求解过程。(7分）

**答：构建哈夫曼树**

**故其哈夫曼编码为：**

**A:10 , B:0000 , C:01 , D:001 , E:11 , F:0001**

1. 给定一棵二叉链表存储的二叉树，试用文字描述判定一棵二叉树是否是完全二叉树的算法基本思想。（7分）

**答：先对二叉树按入队规则先左子树后右子树进行广度优先遍历，并且再遍历过程中按出队顺序由1开始进行连续编号。之后再对该二叉树继续进行一次广度优先遍历，遍历过程中若出现1，当前出队的结点存在左结点，且左结点的编号不是当前结点编号的两倍；2，当前出队的结点存在右结点，且右结点的编号不是当前结点编号的两倍+1时判定该二叉树不是完全二叉树，若遍历完成后均符合要求，则证明该二叉树为完全二叉树。**

5. 已知一棵完全二叉树共有67 个结点，试求: （7分）

(1) 树的深度；

(2) 度为1的结点数；

(3) 叶子结点数；

答：

1. **设树的深度为h，由完全二叉树的性质可知，代入可得其深度为7。**
2. **因为是完全二叉树，故度为1的结点数只能是0或者1。假设该二叉树度为1的结点数为1。因为，又根据二叉树的性质有，矛盾。故该二叉树度为1的结点数只能为0。**
3. **因为，且，故该二叉树叶子结点数为34。**
4. 对给定的一组关键字序列（29,18,25,47,58,12,51,10）, 写出用归并排序方法进行排序的变化过程。（5分）

**答：**

**第一趟：18，29，25，47，12，58，10，51**

**第二趟：18，25，29，47，10，12，51，58**

**第三趟：10，12，18，25，29，47，51，58**

1. **算法填空（共2小题，每空2分，共20分**）

1. 若二叉排序树T中存在其关键字等于 key 的数据元素时，则下面算法删除该数据元素结点，并返回TRUE；否则返回FALSE。请在\_\_\_\_\_\_\_\_处填上适当内容，使其成为一个完整算法。

typedef struct BiTNode {

TElemType data;

struct BiTNode \*lchild, \*rchild;

} BiTNode, \*BiTree;

Status DeleteBST(BiTree &T, KeyType key) {

if (!T) return FALSE;

else {

if (EQ(key, T->data.key)) return Delete(T);

else if (LT(key, T->data.key)) return DeleteBST(T->lchild, key);

else return DeleteBST(T->rchild, key);

}

}

Status Delete(BiTree &p) {

BiTree q, s;

if (!p->rchild) {

q = p; **(1)p=p->lchild** free(q);

}

else if (!p->lchild) {

q = p; **(2)p=p->rchild**  free(q);

}

else {

q = p; **(3)s=p->lchild**  ;

while ( **(4)s->rchild!=NULL**  ) { q = s; s = s->rchild; }

p->data = s->data;

if ( q != p) q->rchild = s->lchild;

else  **(5)q->rchild=NULL**

free(s);

}

return TRUE;

}

2. 下面是快速排序算法。请在\_\_\_\_\_\_\_\_\_\_处填上适当内容，使其成为一个完整算法。

#define Maxsize 100

typedef int keytype;

typedef struct

{ keytype key;

Infotype otherinfo;

}RedType;

typedef struct

{ RedType r[Maxsize+1];

int length;

}SqList;

void Qsort(SqList &L, int low, int high)

{ if (low<high)

{ pivotloc=Partition(L, low, high);

**(6)Qsort(L,low,pivotloc-1**  ;

Qsort(L, pivotloc+1, high);

}

}

int Partition(SqList &L,int low,int high) {

L.r[0]=L.r[low];

pivotkey=L.r[low].key;

while ( low<high ) {

while (low<high&&L.r[high].key>=pivotkey)  **(7)high--**  ;

L.r[low]=L.r[high];

while (low<high&&L.r[low].key<=pivotkey)  **(8)low++**  ;

L.r[high]=L.r[low];

}

L.r[low]= **(9)pivotkey**  ;

return  **(10)low**  ;

}

**六．编写算法（30分）**

1. 试编写算法，从大到小输出二叉排序树中所有的值不小于x的关键字。(10分）

**typedef struct bitTree{**

**elemType data;**

**bitTree \*lchild, \*rchild;**

**};**

**void printBt(bitTree &p){**

**if(!p) return;**

**if(p->data < x) printBt(p->rchild);**

**else{**

**printBt(p->rchild); //遍历右子树**

**print(p->data); //打印关键字**

**printBt(p->lchild); //遍历左子树**

**}**

**}**

2. 设有一个由正整数组成的无序单链表，试编写算法实现下列功能：（10分）

1. 找出最小值结点，并输出该数值；
2. 若该最小值是奇数，则将其与直接后继结点的数值交换；如该最小值是偶数，则将其直接后继结点删除。

**typedef struct ListLink{**

**elemType data;**

**ListLink \*next;**

**};**

**void solve(ListLink &L){**

**ListLink \*p = L; //L带头结点**

**if(!p) return;**

**ListLink \*r = p->next;**

**elemType min = r->data;**

**while(p->next!=NULL){**

**p = p->next;**

**if(p->data < min){**

**min = p->data;**

**r = p;**

**}**

**}**

**printf(r->data);**

**if(r->data & 1){**

**if(r->next!=NULL){**

**elemType t = r->next->data;**

**r->next->data = r->data;**

**r->data = t;**

**}**

**}**

**else{**

**if(r->next!=NULL){**

**p = r->next;**

**r->next = r->next->next;**

**free(p);**

**}**

**}**

**}**

3. 编写一个算法根据用户输入的偶对（以输入0表示结束）建立其有向图的邻接表（设有n个顶点）。（10分）

**typedef struct LinkNode{**

**int v;**

**LinkNode \*next;**

**};**

**typedef struct ListLink{**

**LinkNode \*list;**

**LinkNode \*tail;**

**};**

**void solve(ListLink L[], int n){**

**for(int i = 1; i <= n; i++){**

**L[i].list = (LinkNode \*)malloc(sizeof(LinkNode));**

**L[i].list->next = NULL;**

**L[i].tail = L[i].list;**

**}**

**int v, u;**

**while(scanf("%d", &u)){**

**if(u!=0) scanf("%d", &v);**

**else break;**

**LinkNode \*p = (LinkNode \*)malloc(sizeof(LinkNode));**

**p->v = v;**

**p->next = NULL;**

**L[u].tail->next = p;**

**L[u].tail = p;**

**}**

**}**